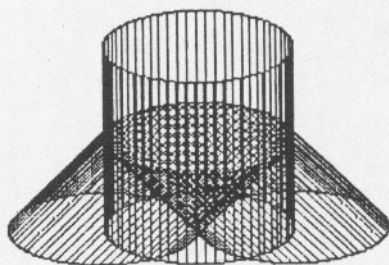
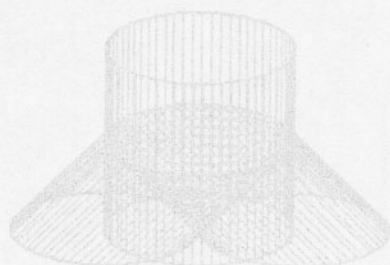
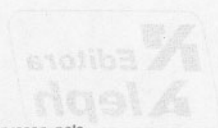
HOTBIT 3ª EDIÇÃO EXPERT

COLEÇÃO DE PROGRAMAS PARA

VOL. I

COLEÇÃO DE PROGRAMAS PARA

3º EDIÇÃO

VOL.1

Este livro foi impresso pela
artes gráficas guaru s/a.
Rod. Presidente Dutra, km 214
Fone: 912-1388 - Guarulhos

COLEÇÃO DE PROGRAMAS PARA

COORDENAÇÃO:

Renato da Silva Oliveira

CO-AUTORES:

Aldo Barduco Jr.

Ligia Neves dos Santos

Luiz Tarcísio de Carvalho Jr.

Milton Maldonado Jr.

Pierluigi Piazzi

Renato da Silva Oliveira

Rubens Pereira Silva Jr.

1986

© 1986 EDITORA ALEPH

Coordenação Editorial.....PIERLUIGI PIAZZI

Coordenação Didática......BETTY FROMER PIAZZI

Editoração e Diagramação..GLAUTER F. MIKAHIL

Copy-desk.................REGINA B. ASSUMPÇÃO

Arte......................ANA LÚCIA ANTICO

Capa e Ilustrações........FERNANDO MORETTI

Produção..................ROSA K. FROMER

ALEPH Publicações e Ass. Ped. Ltda.
Av. Brig. Faria Lima, 1451 - Conj. 31
01451 - São Paulo - SP
Tel.: (011) 212-4917
CAIXA POSTAL: 20.707

Dados de Catalogação na Publicação (CIP) Internacional
(Câmara Brasileira do Livro, SP, Brasil)

Oliveira, Renato da Silva, 1960-
O95c Coleção de programas para o MSX / Renato da Silva Oliveira. -- São Paulo : Aleph, 1986.

1. BASIC (Linguagem de programação para computadores) 2. MSX (Computadores) 3. Programas de computador I. Título.

CDD-001.6425
-001.64
-001.6424

86-0034

Índices para catálogo sistemático:

1. BASIC : Linguagem de programação : Computadores : Processamento de dados 001.6424
2. Computadores : Programas : Processamento de dados 001.6425
3. MSX : Computadores : Processamento de dados 001.64
4. Programas : Computadores : Processamento de dados 001.6425

SUMÁRIO

DESENHOS

MÚSICAS E SONS

NOTA DO EDITOR

Um dos grandes erros que foram cometidos (e ainda estão sendo!) com os usuários de microcomputadores pessoais, consiste em se achar que ensinar BASIC é ensinar computação!

Isso seria como querer ensinar o uso de uma geladeira começando pelo Segundo Princípio da Termo-Dinâmica.

O BASIC, na realidade, é uma linguagem e como tal deve ser ensinada. Para tanto, são muito mais válidas as técnicas didáticas utilizadas no aprendizado de LINGUAS do que, por exemplo, de MATEMÁTICA.

Quem já tentou (e conseguiu) aprender uma língua estrangeira sabe muito bem que é absurdamente mais importante o USO do que o ESTUDO.

Estamos cansados de ver pessoas que, vivendo uns dois meses nos ESTADOS UNIDOS, conseguem dominar o inglês enquanto outras, após anos de estudo na escola, não conseguem nem sequer ler as manchetes do "TIME".

A grande e fundamental diferença entre as duas situações está na interação constante decorrente do uso. Se eu falo errado, não sou entendido; se não entendo direito, faço papel de bobo, a consequência de meus erros é imediata!

A revolução provocada pelos baixos custos da eletrônica digital fez com que o microcomputador pudesse ser utilizado a nível pessoal, permitindo a interação direta e imediata do usuário com a máquina.

Essa revolução permite que o usuário converse em BASIC com o computador e receba respostas instantâneas: se a frase não foi bem construída a máquina acusa um erro de sintaxe. Se a lógica da sequência de instruções tem falhas, coisas estranhas e inesperadas começam a acontecer.

A busca do porquê do erro e sua correção se

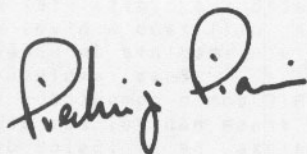
constitui no mais precioso recurso didático para aprendizagem do BASIC.

Com base nestas descobertas feitas ao longo de anos dedicados ao ensino, derrubando idéias preconcebidas, aposentando técnicas ineficientes, é que chegamos à concepção deste livro, dirigido especificamente ao usuário que não conhece computação e quer aprender a conversar com seu computador.

Nele são apresentados programas simples e curtos que o usuário deve digitar (mesmo sem compreendê-los, numa primeira abordagem).

Para que esta digitação seja eficiente, o leitor deve ler com muito cuidado os dois capítulos iniciais: "Digitação e Edição" e "Gravação em Fita".

Após digitar e rodar algumas vezes o programa, para ver que efeitos ele causa no computador, o usuário deve efetuar as alterações sugeridas e fazer suas próprias experiências. Neste momento é que começa o verdadeiro processo de aprendizagem.

Ao completar este volume, o leitor, além de ter um valioso acervo de programas úteis e divertidos, já gravados em fita, terá aprendido de maneira intuitiva a conversar em BASIC com seu MSX.

No aprendizado de uma língua, a consulta aos dicionários e às gramáticas deve ser feita sempre numa fase posterior à da conversação.

Analogamente, após se divertir dialogando com seu micro ao longo deste livro, aconselhamos a consulta a obras mais sistemáticas para tomar conhecimento da "gramática" dos comandos e das funções.

Para tanto, entre outros livros, recomendamos a leitura do LINGUAGEM BASIC MSX desta editora, onde estão organizados, em ordem alfabética, as palavras deste poderoso BASIC, num verdadeiro dicionário enciclopédico.

De qualquer forma, todos os programas deste livro são comentados ressaltando-se suas particularidades e evidenciando-se as técnicas empregadas.

Este é um livro que deverá ser lido, no mínimo, duas vezes: na primeira, o leitor se divertirá aprendendo. Na segunda, já mais consciente, e tendo incorporado um certo vocabulário e algumas técnicas, ele aprenderá se divertindo.

DIGITAÇÃO E EDIÇÃO

Assim que você liga o computador, surgem automaticamente, na tela, duas mensagens (fig. 1.1).

```
Sistema MSX

versão 1.0 Br

G r a d i e n t e
```

```
MSX BASIC versão 1.0 Br
G r a d i e n t e
28815 bytes livres
Ok
█
```

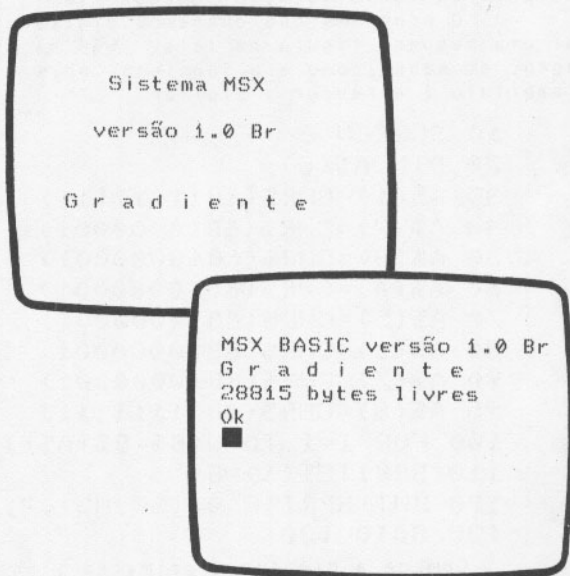



fig. 1.1

Para eliminá-las, de maneira a ter a tela inteiramente à sua disposição para começar a introduzir seu programa, você deve digitar ao mesmo tempo as teclas SHIFT e HOME/ CLS (fig. 1.2). Isso faz com que a tela fique limpa e o cursor se posicione no alto, à esquerda .

fig. 1.2

Acompanhe agora, passo a passo, a digitação de um programa curto. Siga rigorosamente as instruções, prestando bastante atenção em cada item.

O programa que queremos digitar (fig 1.3) desenha uma pequena figura na tela. Não se preocupe, por enquanto, em saber como ele funciona, pois o importante neste capítulo é aprender a digitar.

```
10 SCREEN 2
20 DIM A$(8)
30 A$(1)=CHR$(&B11111111)
40 A$(2)=CHR$(&B10000001)
50 A$(3)=CHR$(&B10000001)
60 A$(4)=CHR$(&B10000001)
70 A$(5)=CHR$(&B10000001)
80 A$(6)=CHR$(&B10000001)
90 A$(7)=CHR$(&B10000001)
95 A$(8)=CHR$(&B11111111)
100 FOR I=1 TO 8:B$=B$+A$(I):NEXT I
110 SPRITE$(1)=B$
120 PUT SPRITE 0,(50,85),2,1
130 GOTO 130
```

fig. 1.3

Comece a digitar a primeira linha e veja o cursor se movendo enquanto vai deixando atrás de si o que você teclou (fig 1.4). Antes de começar, aperte (uma única vez) CAPS LOCK (trava de maiúsculas). Isso faz com que a tela fique o mais parecida possível com a listagem impressa neste livro.

Se você errar alguma letra, basta apertar BS (back space) e o cursor voltará para trás apagando.

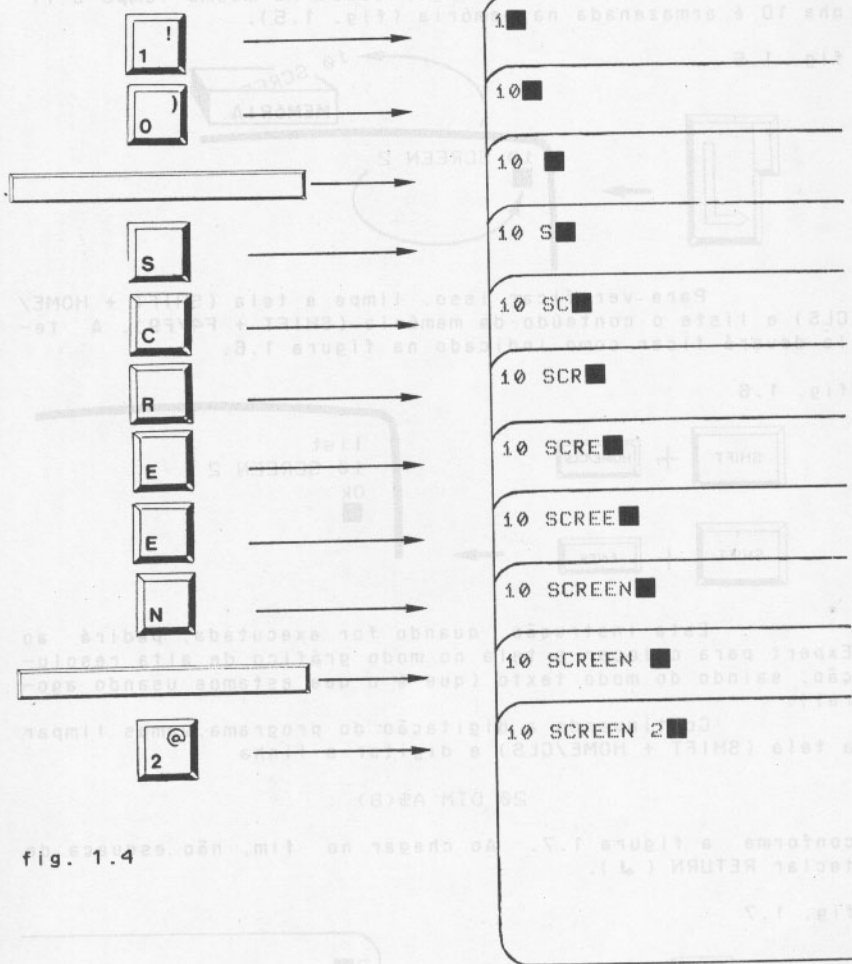



fig. 1.4

Ao terminar a digitação da linha

10 SCREEN 2

você vê o cursor, ainda na frente do último caractere, à espera da introdução de mais letras ou números. Para informar ao seu Expert que esta linha terminou, você deve teclar RETURN (↵).

Neste momento o cursor pula para a próxima linha da tela, à espera de mais instruções. Ao mesmo tempo a linha 10 é armazenada na memória (fig. 1.5).

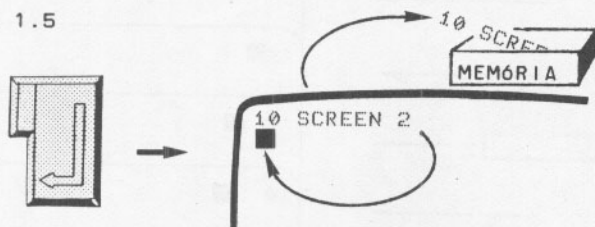
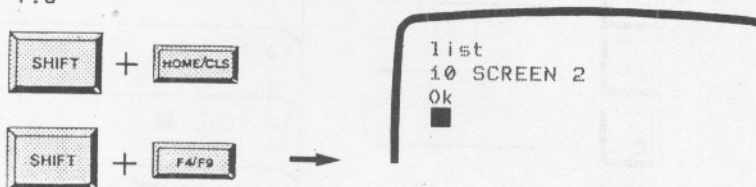
fig. 1.5

Para verificar isso, limpe a tela (SHIFT + HOME/CLS) e liste o conteúdo da memória (SHIFT + F4/F9). A tela deverá ficar como indicado na figura 1.6.

fig. 1.6

Esta instrução, quando for executada, pedirá ao Expert para colocar a tela no modo gráfico de alta resolução, saindo do modo texto (que é o que estamos usando agora!).

Continuando a digitação do programa, vamos limpar a tela (SHIFT + HOME/CLS) e digitar a linha

20 DIM A$(8)

conforme a figura 1.7. Ao chegar no fim, não esqueça de teclar RETURN (↵).

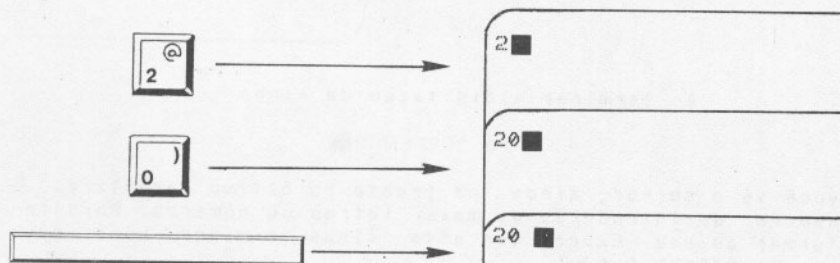
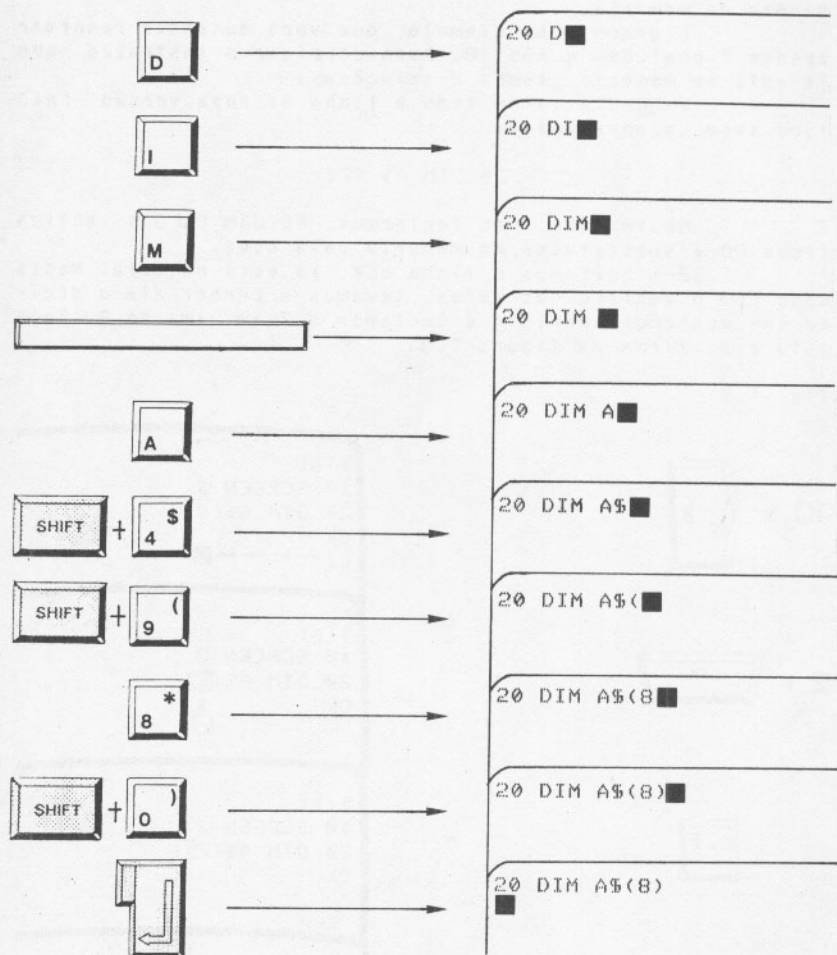
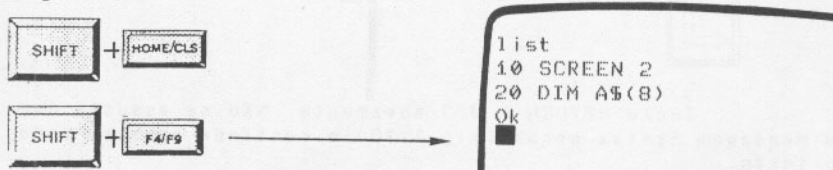
fig. 1.7

Limpe a tela novamente e liste o conteúdo da memória. Agora você terá o que está indicando a figura 1.8.

fig. 1.8

Com a instrução desta linha você reservou 8 posições de memória.

Digamos, por exemplo, que você quisesse reservar apenas 7 posições e não 8. Para corrigir a instrução que já está na memória, temos 2 soluções :

1- Redigitamos toda a linha na nova versão (não faça isto, apenas leia):

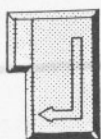
20 DIM A$ (7)

Neste caso, ao teclarmos RETURN (↵) a antiga linha 20 é substituída, na memória, pela nova.

2- Editamos a linha que já está na tela. Neste caso com o auxílio das setas, levamos o cursor até o dígito que queremos corrigir e teclamos o 7 em cima do 8. Faça isto e confirme na figura 1.9.

fig. 1.9

Tecle RETURN (↵) novamente. Não se assuste com a mensagem Syntax error (fig 1.10) e continue acompanhando o texto.

fig. 1.10

Digamos agora que você queira alterar a linha

20 DIM A$(7) para 20 DIM A$(78)

Neste caso, você levará o cursor até o parênteses que está sendo fechado e teclará INSERT (acompanhe o processo na fig. 1.11). O cursor ficará pela metade, indicando que o próximo caractere a ser digitado não se sobreporá ao que já está no local, mas sim o deslocará para a direita, inserindo-se no espaço assim aberto. Digite o 8 que você queria inserir e logo em seguida tecle RETURN(↵).

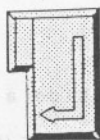
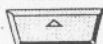
fig. 1.11

11 × ▷
4 × △

```
list
10 SCREEN 2
20 DIM A$(7)
Ok
Syntax error
Ok
```

INSERT

```
list
10 SCREEN 2
20 DIM A$(7)
Ok
Syntax error
Ok
```

8 *

```
list
10 SCREEN 2
20 DIM A$(78)
Ok
Syntax error
Ok
```

↵

```
list
10 SCREEN 2
20 DIM A$(78)
Ok
Syntax error
Ok
```

Vamos listar o conteúdo da memória? (SHIFT+F4) Deverá aparecer o indicado na figura 1.12.

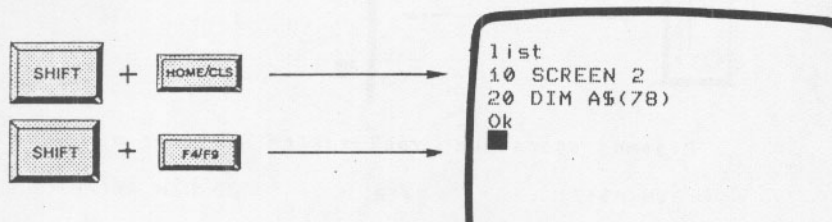
fig. 1.12

Vamos voltar agora à instrução original, reservando apenas 8 posições. Leve o cursor até o caractere 7 e digite DELETE (fig. 1.13).

fig. 1.13

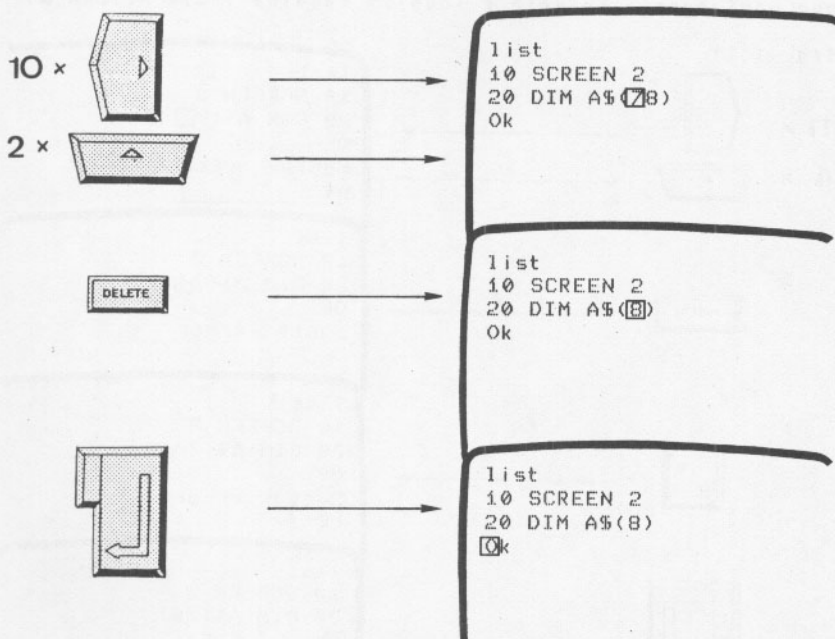
10 ×

2 ×

DELETE

```
list
10 SCREEN 2
20 DIM A$(78)
Ok
```

```
list
10 SCREEN 2
20 DIM A$(8)
Ok
```

```
list
10 SCREEN 2
20 DIM A$(8)
Ok
```

Nesta altura, você já deve ter percebido a diferença entre apagar com BS e com DELETE.

Caso não tenha, limpe a tela (SHIFT+HOME/CLS) e digite 123456789.

A seguir, tecle RETURN (↵) e novamente a sequência 123456789 (seguida de outro RETURN).

A tela ficará como indicado na figura 1.14.

fig. 1.14

123456789
Syntax error
Ok
123456789
Syntax error
Ok

Leve o cursor até o 5 da primeira sequência e digite BS quatro vezes (fig. 1. 15). Leve agora o cursor até o 5 da segunda sequência e tecle DELETE quatro vezes (fig. 1.16).

fig. 1.15

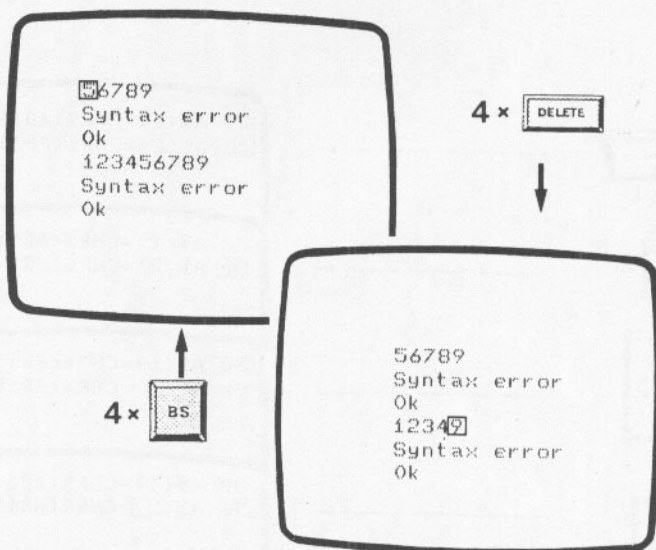



fig. 1.16

Agora que você percebeu a diferença, limpe a tela (SHIFT+HOME/CLS) e vamos continuar a digitação.

Digite a linha 30 (não esqueça o RETURN no final!) e a linha 40 (e RETURN!). A tela terá agora o aspecto mostrado na figura 1.17.

fig. 1.17

Olhando para o programa original (fig. 1.3) verificamos que a linha 50 é praticamente igual à 40. Para poupar trabalho de digitação, podemos usar o seguinte truque: subimos o cursor até a linha 40 (acompanhe pela figura 1.18), substituímos o 4 pelo 5 e deslocamos o cursor até o 2, substituindo-o por um 3. Ao teclarmos RETURN esta nova linha é incorporada na memória sem que a 40 seja apagada.

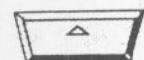
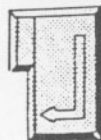
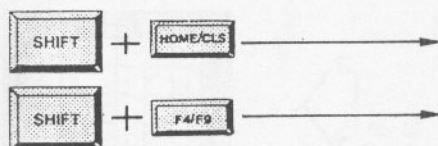
fig. 1.18

```
30 A$(1)=CHR$(&B11111111)
40 A$(2)=CHR$(&B10000001)

5
30 A$(1)=CHR$(&B11111111)
50 A$(2)=CHR$(&B10000001)

5×
30 A$(1)=CHR$(&B11111111)
50 A$(2)=CHR$(&B10000001)

3
30 A$(1)=CHR$(&B11111111)
50 A$(3)=CHR$(&B10000001)

30 A$(1)=CHR$(&B11111111)
50 A$(3)=CHR$(&B10000001)
```

Para se certificar disso, limpe a tela e liste o conteúdo da memória (fig. 1.19).

fig. 1.19

Usando essa listagem você pode gerar o resto do programa; as linhas 60, 70, 80 e 90 podem ser criadas a partir da linha 50 (não esqueça o RETURN para colocar cada nova linha na memória).

A linha 95 pode ser criada a partir da linha 30 e as outras devem, infelizmente, ser digitadas por inteiro. Não se preocupe com a ordem com a qual as linhas são inseridas na memória. O EXPERT se encarrega de arquivá-las em ordem numérica crescente. Ao terminar a digitação, limpe a tela e liste a memória.

A tela deverá mostrar o programa igualzinho à figura 1.3.

Confira com cuidado pois cada vírgula é fundamental para quc o programa funcione.

Se houver necessidade de correções, use todos os truques aprendidos até aqui mas não se esqueça de teclar RETURN para inserir uma linha nova ou alterada na memória.

Limpe a tela (SHIFT+HOME/CLS) e vamos rodar o programa. Para isto basta teclar F5.

A tela se põe no modo gráfico e aparece o quadrado definido pelos dígitos 1 e 0 das variáveis A$ (fig. 1.20).

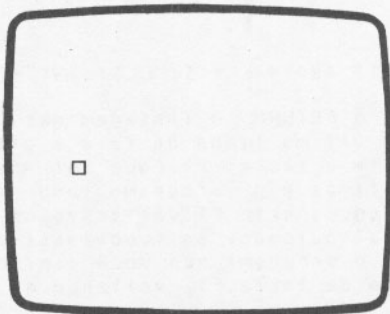
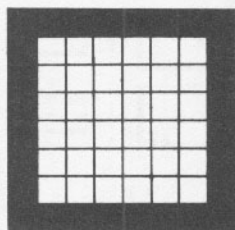
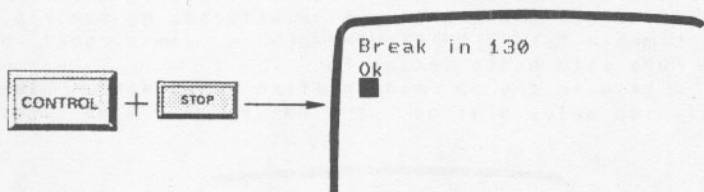
fig. 1.20

Onde há um dígito 1, o ponto é aceso; onde há um dígito 0, ele é deixado apagado (fig. 1.21).

fig. 1.21

```
A$(1) = 1 1 1 1 1 1 1 1
A$(2) = 1 0 0 0 0 0 0 1
A$(3) = 1 0 0 0 0 0 0 1
A$(4) = 1 0 0 0 0 0 0 1
A$(5) = 1 0 0 0 0 0 0 1
A$(6) = 1 0 0 0 0 0 0 1
A$(7) = 1 0 0 0 0 0 0 1
A$(8) = 1 1 1 1 1 1 1 1
```

Digamos que você queira alterar o desenho. Para isto basta listar o programa novamente e alterar alguns 1 e 0.

Antes, porém, vamos aprender mais um truque muito útil para quem tem todos os programas deste livro pela frente. Você deve ter reparado quantas vezes limpamos a tela e listamos o programa. Toda vez que uma operação é muito repetitiva, é conveniente atribuí-la a uma tecla de função, para poupar trabalho.

Vejamos como fazer isso. Inicialmente digite CONTROL+STOP para interromper o programa. A tela assumirá o aspecto da figura 1.22.

fig. 1.22

Digite agora: KEY 1,"CLS:LIST"+CHR$(13)

Após o RETURN a listagem das chaves de função que aparece na última linha da tela é alterada (fig.1.23).

Aperte a tecla F1 (que foi redefinida) e você terá a tela limpa e o programa todo listado. Se quiser brincar um pouco, aperte F1 várias vezes.

Agora cuidado: se você desligar o computador, além de perder o programa que você digitou até aqui, perderá o conteúdo da tecla F1, voltando ao "color" que havia originalmente.

fig. 1.23

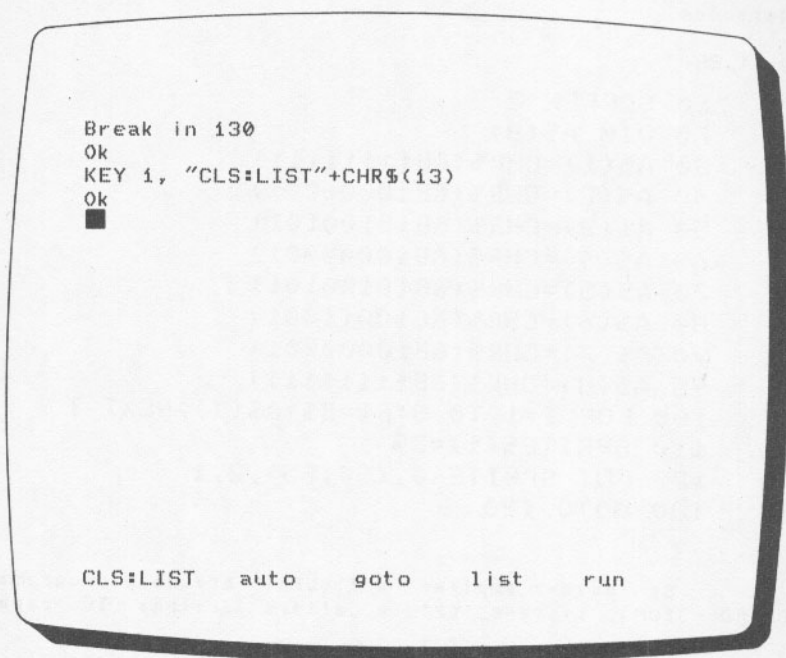
```
Break in 130
Ok
KEY 1, "CLS:LIST"+CHR$(13)
Ok
█








CLS:LIST   auto    goto    list    run
```

Altere agora o programa mudando 1 e 0 de maneira a obter o indicado na figura 1.24. Lembre-se que as alterações somente são registradas na memória se a tecla RETURN (↲) for acionada!

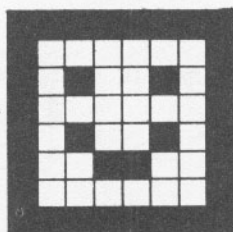
fig. 1.24

Para certificar-se disso, tecle F1 e verifique se suas alterações coincidem com a figura 1.25. Tecle F5 (RUN) para acionar o programa e veja o efeito de suas alterações.

fig. 1.25

```
10 SCREEN 2
20 DIM A$(8)
30 A$(1)=CHR$(&B11111111)
40 A$(2)=CHR$(&B10000001)
50 A$(3)=CHR$(&B10100101)
60 A$(4)=CHR$(&B10000001)
70 A$(5)=CHR$(&B10100101)
80 A$(6)=CHR$(&B10011001)
90 A$(7)=CHR$(&B10000001)
95 A$(8)=CHR$(&B11111111)
100 FOR I=1 TO 8:B$=B$+A$(I):NEXT I
110 SPRITE$(1)=B$
120 PUT SPRITE 0,(50,85),2,1
130 GOTO 120
```

Se quiser ampliar a figura, breque o programa (CONTROL+STOP), liste-o (F1) e altere a linha 10 para

```
10 SCREEN 2,1
```

Limpe a tela (SHIFT + HOME/CLS) e rode-o de novo digitando F5.

Agora você pode brincar à vontade, montando a figurinha que quiser e posicionando-a onde quiser.

Para alterar a figurinha, basta alterar a sequência de 1 e 0 , mantendo-se, porém, dentro de uma tabela 8x8.

A posição e a cor da figurinha são dadas pela linha 120 (fig. 1.26).

fig. 1.26

Uma alteração gerando efeitos malucos seria:

```
120 PUT SPRITE 0,(255*RND(2),191*RND(2)),13*RND(2)+2,1
```

Faça-a e veja seus efeitos!

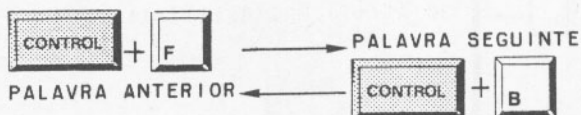
Antes de partir para a digitação de outros programas, porém, seria conveniente que você aprendesse mais alguns truques.

Liste o programa e coloque o cursor no começo (basta digitar HOME/CLS, sem SHIFT!).

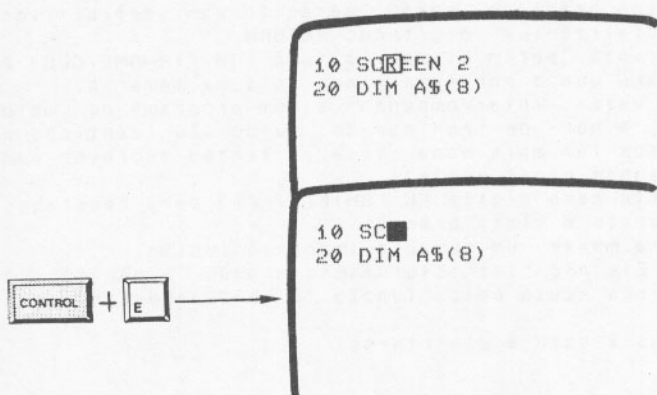
Digite agora CONTROL+F : o cursor se posiciona no início da palavra seguinte!

Faça isso várias vezes e veja que o EXPERT entende por "palavra seguinte" a que começa por letra ou algarismo.

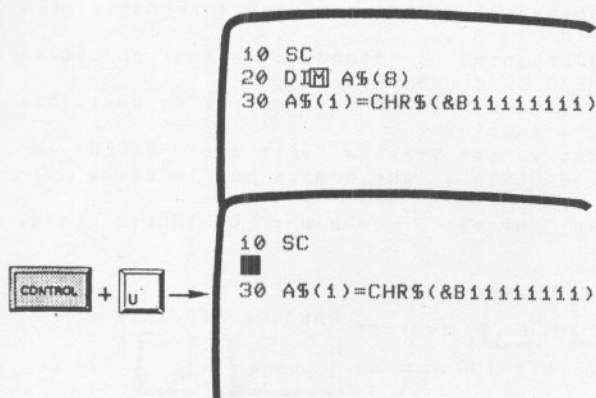
O caminho inverso é dado por CONTROL+B (fig. 1.27).

fig. 1.27

Liste o programa (F1) e coloque o cursor no começo (HOME ou CONTROL+K).

Desloque o cursor até o meio da linha 10 (em cima do R, por exemplo) e digite CONTROL+E (fig. 1.28).

fig. 1.28

Esse comando apaga desde o cursor até o fim da linha de programa!

Um que tem efeitos mais drásticos é o CONTROL+U. Coloque o cursor sobre o M da linha 20 e dê esse comando (fig. 1.29).

fig. 1.29

Outro comando útil é o CONTROL+N: ele leva o cursor até o fim da linha de BASIC, permitindo uma boa economia de tempo em certos casos.

Coloque novamente o cursor em cima (HOME) e comece a apertar a tecla TAB (ou CONTROL+I, tanto faz!). O cursor começa a andar de 8 em 8 posições arrasando o que está no caminho.

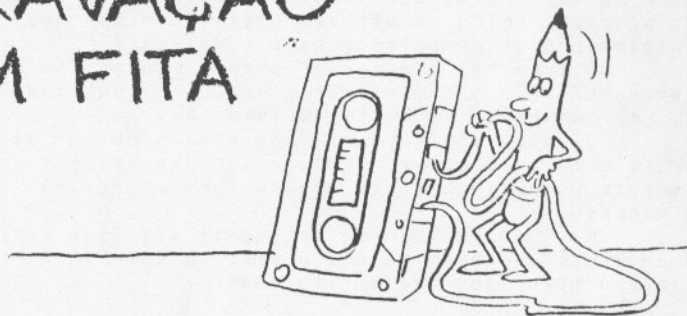
Estes estragos, porém, só se tornam definitivos se você "oficializá-los" digitando RETURN.

Se você, porém, limpar a tela (SHIFT+HOME/CLS) e teclar F1, verá que o programa ainda está na memória.

Às vezes, interrompendo algum programa no meio da execução, a cor de frente e de fundo são idênticas e não conseguimos ler mais nada (você já tentou escrever com tinta verde sobre papel verde?).

Neste caso digite F6 (SHIFT + F1) para reestabelecer fundo preto e tinta branca!

Finalmente, um comando importantíssimo: CONTROL+G . Ele não faz absolutamente nada a não ser um "beep" sem graça cuja única função é chatear parentes e vizinhos!

Mãos à obra e divirta-se!

O EXPERT possui dois tipos de memórias: a ROM e a RAM.

A ROM já vem com todo o seu conteúdo gravado de fábrica e nela não podem ser nem acrescentadas nem eliminadas informações.

Na RAM, em compensação, podemos escrever e apagar à vontade. É como se a ROM fosse um livro já impresso e a RAM um quadro negro no qual podemos usar giz e apagador.

Quando desligamos o micro, o conteúdo da RAM, onde o usuário escreve seus programas em BASIC, é apagado enquanto que o da ROM é preservado.

Para arquivar um programa escrito na RAM, você deve então gravá-lo. Se no seu EXPERT estiver conectado um "disk-drive" (sistema de disco magnético), basta digitar

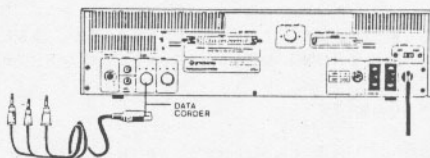
```
SAVE "nome do programa"
```

e ele será gravado no disco com o título escolhido.

Se você quiser gravá-lo em fita cassete, deverá tomar alguns cuidados.

CONEXÃO DE CABOS

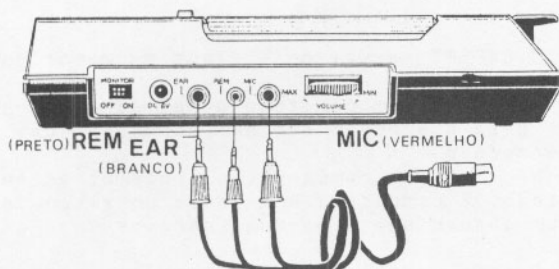
Ligue o cabo do gravador na parte traseira do console (conexão DATA-CORDER) conforme a fig. 2.1.

fig. 2.1

No outro terminal do cabo você tem 3 plugs (fig. 2.2). O vermelho deve ser conectado na entrada do microfone do gravador (MIC). É através deste terminal que passam as informações do computador para o gravador.

O terminal branco serve para mandar informações do gravador para o computador e deve ser conectado na entrada EAR (MONITOR, em certos gravadores).

O plug preto (mais fino que os outros dois) é o controle remoto do motor (REM) e não precisa ser obrigatoriamente conectado, pois alguns gravadores não possuem este recurso.

No DATA-CORDER da Gradiente ele deve ser conectado em REM o que permite que o próprio computador ligue e desligue o motor que faz a fita andar.

fig. 2.2

GRAVAÇÃO DO PROGRAMA

Digite este curto programa no Expert:

fig. 2.3

```
10 REM PROGRAMA UM
20 SCREEN 1
30 FOR N = 0 TO 255
40 VPOKE 6144 + N,N
50 NEXT N
60 LOCATE 0,8
```

Se você tiver curiosidade de saber o que ele faz, rode-o com F5 (RUN) algumas vezes e depois limpe a tela (SHIFT+HOME/CLS).

Depois digite:

CSAVE "UM" (sem teclar RETURN!)

Verifique se a fita está bem posicionada e se o trecho inicial (sem material magnético) já foi ultrapassado. Aperte as teclas PLAY e REC (gravação). No DATA-CORDER isso correponde às teclas SAVE e LOAD.

Se o plug REM estiver conectado, a fita ficará parada. Caso contrário, começará a rodar.

Digite RETURN (↵) e a gravação começará.

Ao terminar, aparecerá um Ok na tela. Pare o gravador se o plug REM não estiver conectado (se estiver, ele parará sozinho).

Limpe a tela (SHIFT+HOME/CLS) e liste o programa (SHIFT+F4).

Altere-o, agora, conforme a listagem da figura 2.4.

fig. 2.4

```
10 REM PROGRAMA DOIS
20 SCREEN 1
30 FOR N = 0 TO 255
40 VPOKE 6144 + N,N
45 VPOKE 8192 + N,N
50 NEXT N
60 LOCATE 0,8
```

Se tiver curiosidade, rode-o com F5 (RUN).

Limpe a tela e liste o programa. Seu Expert parece agora um arlequim bêbado. Não se preocupe! Tecle F6 (SHIFT + F1) e a ordem será restabelecida.

Com a fita parada no final do programa UM digite:

CSAVE "DOIS"

e repita todo o procedimento de gravação.

LEITURA DA FITA

Vamos agora recuperar os programas gravados (UM e DOIS).

Rebobine a fita até o começo, limpe a tela e, se quiser, a RAM do computador também (digitando NEW e RETURN ou então desligando o computador).

Digite agora:

CLOAD "DOIS" (sem RETURN aind-,)

Aperte PLAY(ou LOAD no DATA-CORDER) RETURN(↵) no computador, a leitura começará.

Se tudo estiver correto, a leitura passará pelo programa UM mas não o carregará na memória pois você pediu o DOIS. Isto será indicado na tela por

Skip:UM

A seguir, o título (só o título!) do segundo programa será lido e a tela mostrará:

Found:DOIS

Depois disso, o programa propriamente dito será lido, e a tela indicará o fim do processo mostrando Ok.

Liste o programa com F9 (SHIFT+F4) e você deve encontrar o programa da figura 2.4.

PROBLEMAS NA LEITURA

Se o procedimento de leitura descrito não correu conforme o esperado, houve algum problema que vamos agora tentar contornar.

Os gravadores comuns são projetados para reproduzir SOM e não o sinal digitalizado de um computador. Consequentemente, eles devem ser regulados e ajustados. Isto corresponde mais ou menos a afinar um violino: uma vez feito o ajuste, se ninguém mexer, a afinação durará por muito tempo.

1) Verifique se a fita é de boa qualidade. Existem fitas de péssima qualidade que soltam a emulsão magnética (sujando inclusive a cabeça de gravação). Não use fitas de longa duração, sendo ideal a C-20 ou, no máximo, a C-30.

2) Limpe a cabeça de gravação com cotonete embebido em álcool. Deixe secar. Para gravadores com muito uso (30 horas ou mais), convém desmagnetizar o cabeçote (qualquer rádio-técnico faz isso em 5 minutos).

3) Verifique se o cabo não está com defeito.

4) Evite qualquer vizinhança com fontes de poluição magnética: imãs, bobinas, transformadores, traseiras de televisão, reatores de lâmpadas fluorescentes, motores elétricos, emissoras de FM, etc.

5) Faça uma boa gravação, repetindo os procedimentos indicados neste capítulo, e tente ler os dados mudando o ajuste de volume e tonalidade (se existir este controle no seu gravador). Tenha paciência e tente todas as combinações possíveis.

Assim que conseguir a leitura anote os valores ótimos ou, melhor ainda, faça uma marca nos controles.

6) Se tudo isto não funcionar, troque o gravador, pois existem alguns modelos que só são compatíveis com computadores após sérias modificações de circuito. Lembre-se

que uma boa reprodução de música não tem nada a ver com gravação de bits.

7) Use, para leitura, o mesmo gravador usado para gravação. Ligeiras mudanças de rotação ou na regulagem do azimute (ângulo de alinhamento da cabeça de gravação) podem incompatibilizar a gravação com a leitura.

LEITURA DE FITAS DE TERCEIROS

Quando você tem uma fita que foi gravada num gravador diferente do seu, podem ocorrer problemas de leitura.

O mais fácil de ser resolvido é o que se refere ao volume: às vezes você deve procurar um ajuste diferente do usual.

O outro problema é do alinhamento do cabeçote.

Verifique o ajuste do azimute da cabeça que é feito por um pequeno parafuso do lado esquerdo (figura 2.5). Em alguns gravadores já existe um furo para alcançá-lo com uma chave de fenda. Em outros, o furo deve ser aberto com uma furadeira. Ponha o programa para rodar e ouça-o! Girando o parafuso do azimute de um lado para o outro você deve achar um ponto de estridência máxima. Esse é o ponto de ajuste.

fig. 2.5

Lembre-se, porém, que ao regular o gravador para outras fitas, você o está desregulando para as suas.

Um procedimento inteligente seria de não mexer em seu gravador e fazer toda essa procura com outro. Após conseguir ler a fita e carregar o programa no seu computador, grave-o novamente no SEU gravador de maneira a ter uma gravação compatível.

A FITOTECA

Para criar sua coleção de programas é conveniente ser organizado.

1) Faça um caderno registrando as fitas, seu conteúdo e, se possível, a marcação do contador correspondente ao início de cada programa.

2) Use fitas curtas, não só por motivos técnicos (as longas são mais frágeis) mas também para facilitar a busca de um determinado programa.

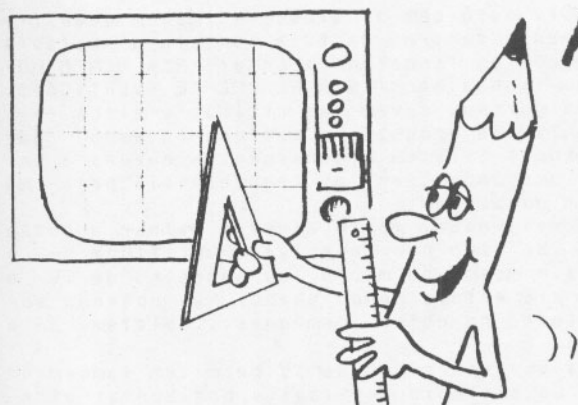
3) Armazene as fitas longe do calor, umidade e campos magnéticos.

4) Tendo uma fita de conteúdo desconhecido, carregue os programas sequencialmente usando o comando

CLOAD

Desta forma, o primeiro programa que for lido será carregado. Identifique-o e continue o procedimento até o fim, de maneira a levantar o índice do conteúdo.

5) Não seja "pão-duro": copie fitas para os amigos (se possível no gravador deles!). Seja, porém, um pirata "gentleman": cite sempre a fonte e o autor do programa.

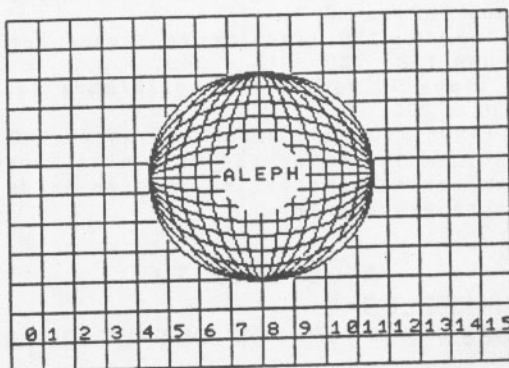
6) Bom divertimento!

AJUSTE DE TV

INSTRUÇÕES

Uma das primeiras coisas que você deve fazer ao ligar um computador num aparelho de TV é ajustá-lo para que a imagem na tela não "canse" a vista. Os monitores de vídeo (aparelhos especiais para uso com computadores) em geral produzem imagens muito boas, tornando desnecessária uma maior preocupação do usuário com seu ajuste. Entretanto, mesmo que você possua um, não perderá nada digitando este programa.

Quando executado, ele gera uma imagem no vídeo e uma sequência de notas musicais.

A imagem possui 13 linhas horizontais e 17 linhas verticais ocupando toda a tela. No centro da tela é construída uma esfera com uma espécie de visor, e na parte inferior são apresentadas as 16 cores com seus respectivos códigos (fig. 3.1).

fig. 3.1

Quando a TV está bem ajustada, a imagem deve ficar fixa e inteiramente dentro da tela (exceto a palavra ALEPH no centro), com as linhas horizontais REALMENTE HORIZONTAIS e com as linhas verticais REALMENTE VERTICAIS. Além disso, os sons gerados devem ser nítidos e altos.

Se a sua TV não produzir o que descrevemos, não chame o técnico ainda! Existem alguns pontos chaves a serem verificados e que podem ser os responsáveis pela má qualidade da imagem ou do som!

Antes de mais nada, tente ajustar melhor o botão de sintonia da TV. Se isso não resolver, verifique se o canal selecionado é o mesmo no micro, no aparelho de TV e no adaptador, caso ele esteja sendo usado. Não obtendo sucesso num canal, tente no outro (em geral, existem dois canais permitidos!).

A maioria dos aparelhos de TV permitem também um ajuste individual da sintonia vertical e horizontal através de botões situados na parte traseira. Use-os apenas como último recurso!

Quando conseguir uma imagem estável, regule o contraste e o brilho de modo a não prejudicar seus olhos. Guarde as posições em que estão os botões para que nas próximas vezes seu trabalho seja minimizado.

Com o aparelho de TV corretamente ajustado, você pode programar sem se cansar!

DIGITAÇÃO

Este programa não envolve nenhuma dificuldade para ser introduzido. Entretanto, se você não possui muita prática com o teclado do MSX, preste bastante atenção!

Como já vimos no primeiro capítulo, você pode economizar tempo na digitação de linhas semelhantes, bastando para isso levar o cursor até a linha já introduzida, mudar seu número e alterar seu conteúdo.

A linha 50 pode servir de base para introdução das linhas 110, 140, 180, 260 e 270, pois todas são aberturas de laços FOR... NEXT.

A linha 100, igualmente, pode ser usada para gerar as linhas 130, 160, 210 e 340.

A linha 90 pode ser transformada nas linhas 120, 150, 220, 230 e 290.

A linha 170 pode gerar as linhas 190, 200 e 240.

As demais linhas são distintas, apesar de algumas possuirem partes semelhantes. Não se perde muito tempo digitando-as separadamente.

```
10 '        Ajuste de TV
20 MAXFILES=1
30 OPEN "GRP:" FOR OUTPUT AS #1
```

```
40 SCREEN 2,1:COLOR 4,7,5:CLS
50 FOR F=0 TO 15
60 G=F*16:H=G+15
70 PRESET(G+5*(G>0),168)
80 PRINT #1,F
90 LINE(G,176)-(H,191),F,BF
100 NEXT F
110 FOR F=0 TO 256 STEP 16
120 LINE(F,0)-(F,192)
130 NEXT F
140 FOR F=0 TO 192 STEP 16
150 LINE(0,F)-(255,F)
160 NEXT F
170 CIRCLE(128,88),58,7:PAINT(128,88),7
180 FOR B=57 TO 4 STEP-8
190 CIRCLE(128,88),57,4,,,57/B
200 CIRCLE(128,88),57,4,,,B/57
210 NEXT B
220 LINE(128,30)-(128,160),4
230 LINE(71,88)-(184,88),4
240 CIRCLE(128,88),20,7:PAINT(128,88),7
250 B$="      ALEPH"
260 FOR F=49 TO 56
270 FOR G=65 TO 71
280 LINE (108,84)-(150,92),7,BF
290 PRESET(110,84):A$=RIGHT$(B$,5)
300 PRINT#1,A$
310 B$=RIGHT$(B$,1)+LEFT$(B$,10)
320 P$="V150"+CHR$(F)+CHR$(G)
330 PLAY P$,P$,P$
340 NEXT G,F
350 GOTO 260
```

ANÁLISE

Vamos agora estudar um pouco esse programa e tentar identificar suas partes principais.

As linhas iniciais, até a 40, apenas preparam a tela para a produção da imagem, definindo a cor da borda, do fundo e do primeiro plano que serão usadas e abrindo um arquivo na tela de alta resolução. Não se preocupe muito com isso por enquanto!

O laço entre as linhas 50 e 100 produz as cores e seus códigos na parte inferior do vídeo.

O laço entre as linhas 110 e 130 produz as linhas verticais, e o laço entre as linhas 140 e 160 produz as linhas horizontais.

A linha 170 limpa uma área circular no centro da tela onde posteriormente o laço entre as linhas 180 e 210 gera uma esfera.

As linhas 220 e 230 completam a figura da esfera no centro da tela e a linha 240 limpa uma outra região circular e menor no seu centro. Dentro dessa região é apresentada, como se estivesse girando, a palavra ALEPH, enquanto as notas musicais vão sendo produzidas.

As notas e a palavra girando são produzidas pelas linhas finais do programa. Por enquanto não é necessário que as estudemos.

Agora que já identificamos as principais partes do programa, vamos fazer algumas experiências com ele. Antes, porém, é conveniente que você o grave se for utilizá-lo posteriormente!

Altere as linhas 110 e 140 colocando STEP 8 ao invés de STEP 16. Isso fará com que aumentem as linhas no vídeo.

Agora mude o STEP-8 da linha 180 para STEP-4. Assim, a esfera produzida terá uma superfície mais detalhada! Uma alteração mais drástica pode ser obtida mudando de 7 para 4 a cor definida na linha 170, e mudando de 4 para 7 a cor definida nas linhas 190, 200, 220 e 230. Com essas alterações, a esfera do centro da tela se degenera.

Como última sugestão, acrescente a linha:

```
165 CIRCLE(128,88),58,7:PAINT(128,88),7
```

Isso faz com que a esfera central seja novamente produzida, porém de uma forma um pouco alterada!

Agora é a sua vez de propor novas alterações!

Tente melhorar o "visual" do programa!

INSTRUÇÕES

Este programa não é um jogo. Nem tampouco um utilitário. Ele é, contudo, um programa criativo!

A palavra CALEIDOSCÓPIO provém do grego CALEIDOS (=do belo) + SCOPIO (=visor), portanto, significa VISOR DO BELO. Isso define para que serve o programa!

Quem já fez um caleidoscópio com papelão vidros coloridos e espelhos, sabe que o número de imagens diferentes que se pode observar através dele é muito grande. Nosso programa, apesar de bastante curto, é capaz de gerar milhões e milhões de figuras diferentes no vídeo do MSX.

Cada figura é formada por quatro partes perfeitamente simétricas, dando-nos a impressão de que a tela é o interior de um caleidoscópio. Você ainda pode participar da confecção da figura digitando qualquer tecla de letras. Experimente para ver!.

DIGITAÇÃO

O programa é bastante curto mas, mesmo assim, se você quiser economizar tempo, tome muito cuidado ao digitar as linhas com várias instruções separadas por dois pontos e utilize as linhas já introduzidas para gerar outras, sempre que possível.

Observe na listagem que a linha 40 pode ser usada para produzir a 50. Esta, por sua vez, serve de base para a linha 70.

A linha 90 pode ser usada para gerar as linhas 100, 110 e 120.

Lembre-se: para usar uma linha já digitada, leve o cursor até ela, altere seu número e seu conteúdo (se necessário) e digite RETURN!

```
10      Caleidoscopio
20 SCREEN 3:C=84:I=127:D=0:E=D
30 FOR G=E TO C*(1+RND(-TIME)*3)
40 H=INT(36*RND(-TIME)):IF H/4<>H\4 THE
N 40
50 D=INT(D+H*RND(-TIME)):IF D/4<>D\4 TH
EN 50
60 E=INT(E+H*RND(-TIME)):IF E/4<>E\4 TH
EN 60
70 D=D AND D<=C:E=E AND E<=C
80 K=INT(2+13*RND(-TIME)):P$="V15S14M40
00L8N"+STR$(INT(RND(-TIME)*60+9)):PLAY
P$
90 PSET(I+D,C+E),K:PSET(I+D,C-E),K
100 PSET(I-D,C+E),K:PSET(I-D,C-E),K
110 PSET(I+E,C+D),K:PSET(I+E,C-D),K
120 PSET(I-E,C+D),K:PSET(I-E,C-D),K
130 IF INKEY$<>"" THEN CLS
140 NEXT G:GOTO 20
```

ANÁLISE

O funcionamento do CALEIDOSCÓPIO é bastante simples. A linha 20 define a tela a ser usada (no caso, a tela gráfica de baixa resolução) e atribui os valores iniciais às variáveis C, I, D e E.

A linha 30 abre um laço que será repetido um número indefinido de vezes.

As linhas entre 40 e 70 servem para gerar dois números aleatórios e inseri-los nas variáveis D e E. Elas serão usadas para produzir a figura no vídeo.

A linha 80 insere um número entre 2 e 14 na variável K e gera uma nota musical ao acaso.

As linhas de 90 à 120 produzem os pontos da figura na tela.

A linha 130 verifica se você está pressionando alguma tecla e, em caso positivo, limpa a tela.

Finalmente, a linha 140 fecha o laço aberto na linha 30 e, após sua última execução, desvia o programa para a linha 20, reiniciando tudo!

Após ter executado o programa algumas vezes e verificar como ele funciona, experimente alterá-lo deixando-o assim:

```
10 '    Caleidoscopio II
20 SCREEN 2:C=84:I=127:D=0:E=D
```

```
30 FOR G=E TO C*(1+RND(-TIME)*3)
40 H=INT(36*RND(-TIME)):IF H/4<>H\4 THE
N 40
50 D=INT(D+H*RND(-TIME)):IF D/4<>D\4 TH
EN 50
60 E=INT(E+H*RND(-TIME)):IF E/4<>E\4 TH
EN 60
70 D=D AND D<=C:E=E AND E<=C
80 P$="V15S14M4000L8N"+STR$(INT(RND(-TI
ME)*60+9)):PLAY P$
90 PSET(I+D,C+E):PSET(I+D,C-E)
100 PSET(I-D,C+E):PSET(I-D,C-E)
110 PSET(I+E,C+D):PSET(I+E,C-D)
120 PSET(I-E,C+D):PSET(I-E,C-D)
130 IF INKEY$<>"" THEN CLS
140 NEXT G:GOTO 20
```

Ou assim:

```
10       Caleidoscopio III
20 SCREEN 2:C=84:I=127:D=0:E=D
30 FOR G=E TO C*(1+RND(-TIME)*3)
40 H=INT(36*RND(-TIME)):IF H/4<>H\4 THE
N 40
50 D=INT(D+H*RND(-TIME)):IF D/4<>D\4 TH
EN 50
60 E=INT(E+H*RND(-TIME)):IF E/4<>E\4 TH
EN 60
70 D=D AND D<=C:E=E AND E<=C
80 P$="V15S14M4000L8N"+STR$(INT(RND(-TI
ME)*60+9)):PLAY P$
90 LINE(I,C)-(I+D,C+E):LINE(I,C)-(I+D,C
-E)
100 LINE(I,C)-(I-D,C+E):LINE(I,C)-(I-D,
C-E)
110 LINE(I,C)-(I+E,C+D):LINE(I,C)-(I+E,
C-D)
120 LINE(I,C)-(I-E,C+D):LINE(I,C)-(I-E,
C-D)
130 IF INKEY$<>"" THEN CLS
140 NEXT G:GOTO 20
```

Ou ainda, assim:

```
10 '     Caleidoscopio IV
20 SCREEN 2:C=84:I=127:D=0:E=D
30 FOR G=E TO C*(1+RND(-TIME)*3)
40 H=INT(36*RND(-TIME)):IF H/4<>H\4 THE
N 40
50 D=INT(D+H*RND(-TIME)):IF D/4<>D\4 TH
EN 50
60 E=INT(E+H*RND(-TIME)):IF E/4<>E\4 TH
EN 60
70 D=D AND D<=C:E=E AND E<=C
80 P$="V15S14M4000L8N"+STR$(INT(RND(-TI
ME)*60+9)):PLAY P$
90 LINE(I+D,C+E)-(I+D,C-E)
91 LINE(I+D,C-E)-(I-D,C-E)
92 LINE(I-D,C-E)-(I-D,C+E)
93 LINE(I-D,C+E)-(I+D,C+E)
94 LINE(I+E,C+D)-(I+E,C-D)
95 LINE(I+E,C-D)-(I-E,C-D)
96 LINE(I-E,C-D)-(I-E,C+D)
97 LINE(I-E,C+D)-(I+E,C+D)
130 IF INKEY$<>"" THEN CLS
140 NEXT G:GOTO 20
```

Agora o efeito não parece mais com um caleidoscópio porém continua a ser belo.

ANAGRAMA

INSTRUÇÕES

Um ANAGRAMA é uma palavra ou frase formada a partir de outra por meio de uma permutação de letras. Por exemplo, a palavra ROMA é um anagrama da palavra AMOR.

O jogo deve ser disputado entre duas pessoas. Cada uma deve tentar descobrir qual a frase original introduzida pela outra a partir de um dos possíveis anagramas apresentados pelo computador.

Apesar do programa aceitar frases de até trinta e cinco letras, é interessante, pelo menos no início, os dois jogadores combinarem a introdução de palavras curtas de até seis ou sete letras. Caso contrário, o anagrama torna-se muito complicado e difícil de ser decifrado.

Quando um dos jogadores estiver decifrando o anagrama do outro, deve introduzir uma letra de cada vez seguida da tecla RETURN. A introdução de um espaço pode ser feita simplesmente pressionando-se RETURN.

Cada letra ou espaço introduzido em lugar incorreto será contada como erro.

Vence o jogador que conseguir descobrir a palavra ou frase original cometendo o menor número de erros.

DIGITAÇÃO

Não há grandes problemas quanto à digitação do programa ANAGRAMA.

Tome apenas cuidado nas linhas 160 e 170 onde aparecem as instruções MID$(B$,Q,1) e MID$(B$,W,1). Elas devem ser digitadas exatamente como estão listadas: sem qualquer espaço entre os caracteres.

```
1 REM "Anagı
10 KEYOFF
20 CLS : COLOR 11,8
30 LOCATE 14,0
40 PRINT "ANAGRAMA"
50 LOCATE 2,4
60 PRINT "Escreva uma frase de no maxim
o 35      letras, sem o outro jogador ve
r."
70 LOCATE 1,12
80 LINE INHUT A$
90 B$=A$
100 C = LEN(A$)
110 IF C > 35 THEN RUN
120 FOR I = 1 TO 3*C
130 Q = INT (RND(-TIME) * C + 1)
140 W = INT (RND(-TIME) * C + 1)
150 Y$ = MID$(B$,Q,1)
160 MID$(B$,Q,1) = MID$(B$,W,1)
170 MID$(B$,W,1) = Y$
180 NEXT I
190 COLOR 11,4 : CLS
200 LOCATE 14,0 : PRINT "ANAGRAMA"
210 LOCATE 2,4 : PRINT "O anagrama e :"
220 Z = INT ((38-C)/2) : FOR I = 0 TO C
+1 : LOCATE Z+I,6 : PRINT CHR$(220); :
LOCATE Z+I,8 : PRINT CHR$(223); : NEXT
I
230 LOCATE Z,7 : PPINT CHR$(221) : LOCA
TE Z+C+1,7 : PRINT CHR$(202)
240 LOCATE Z+1,7 : PRINT B$
250 LOCATE 2,10 : PRINT "Desembaralhe a
s letras ..."
260 FOR I = 0 TO C+1 : LOCATE Z+I,12 :
PRINT CHR$(220); : LOCATE Z+I,14 : PRIN
T CHR$(223); : LOCATE Z+I,13 : PRINT "?
"; : NEXT I
270 LOCATE Z,13 : PRINT CHR$(221) : LOC
ATE Z+C+1,13 : PRINT CHR$(202)
280 I = 1
```

```
290 LOCATE 2,16 : PRINT "Numero de erro
s =";E
300 LOCATE 5,19 : PRINT "Qual a letra
                    " : LOCATE 18,19 : C$="
" : INPUT C$ : C$ = LEFT$(C$,1)
310 IF C$ = "" THEN C$=" "
320 IF C$ = MID$(A$,I,1) THEN 350
330 E = E + 1
340 GOTO 290
350 LOCATE I+Z,13 : PRINT C$
360 I = I + 1
370 IF I <= C THEN 300
380 LOCATE 3,21 : PRINT "APERTE UMA LET
RA PARA RECOMECAR"
390 IF INKEY$ = "" THEN 390
400 RUN
```

ANÁLISE

As linhas de 10 a 80 preparam a tela para a introdução da palavra ou frase por um dos jogadores. O programa pára na linha 80 à espera da frase que deve, após digitada, ser seguida da tecla RETURN. Esta linha pode ser substituída por INPUT A$. Faça-o e note a diferença.

A linha 90 "guarda" o conteúdo da variável A$ (frase introduzida) na variável B$ para que as letras possam ser embaralhadas posteriormente sem que a frase original seja perdida.

A linha 100 armazena o comprimento da frase na variável C e a linha 110 testa se este é maior que 35. Em caso positivo, o programa recomeça.

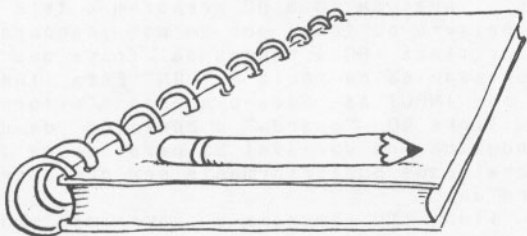
As linhas 120 a 180 executam o embaralhamento das letras. Inicialmente é definido um laço entre as linhas 120 e 180 para que o algoritmo de permutação dos caracteres seja executado três vezes o comprimento (C) da frase introduzida.

O algoritmo em si (linhas 130 a 170) consiste em "sortear" dois números compreendidos entre 1 e C (linhas 130 e 140) e assumi-los como posições na "string" B$. Estas posições são invertidas (linhas de 150 a 170).

A vantagem desse procedimento é produzir uma variável B$ que contenha os mesmos caracteres que a variável A$, permutados e sem repetições.

As linhas de 190 a 300 preparam a tela que apresenta o anagrama (variável B$), o número de erros (E), e pede a introdução das letras para a decifração da frase.

As linhas de 310 a 370 testam se a letra introduzida (armazenada na variável C$) se encaixa no lugar correto (linha 320). Caso positivo, a letra é impressa (linha 350), o contador de posições (I) é incrementado (linha 360) e se a última posição não tiver sido atingida o programa pergunta pela próxima letra (linha 370). No caso da letra introduzida não se encaixar no lugar correto, é incrementado o contador de erros (linha 330) e o programa é desviado para a linha 290 para que o número de erros seja impresso e novamente seja pedida a introdução de uma letra.

Finalmente as linhas 380 e 390 estabelecem a espera da pressão de uma tecla para o programa recomeçar (linha 400).

INSTRUÇÕES

Um TWO-LINER é um programa com apenas duas linhas.

Isso não é nada difícil de ser feito. Veja o programa a seguir:

```
10 INPUT"Qual o seu nome",n$
20 PRINT n$: GOTO 10
```

Ele é um TWO-LINER, porém extremamente patético!

Fazer um TWO-LINER é fácil!

Difícil é conseguir que ele gere efeitos sofisticados no micro!

Os programas apresentados mais adiante são TWO LINERS de primeira categoria, gerando figuras complexas e sofisticadas no vídeo.

Há fundamentalmente tres parâmetros que permitem julgar a qualidade de um programa:

* O resultado final obtido através dele;
* A quantidade de memória que ele utiliza e
* O tempo que ele leva para ser executado.

Os resultados finais dos TWO-LINERS apresentados a seguir, você mesmo poderá julgar!

A memória ocupada por cada um deles é, certamente, menor que 1/2 kbyte (512 bytes). Na verdade, a maioria deles ocupa bem menos que isso (por volta de 256 bytes!).

O tempo de execução é que, às vezes, se torna demasiado longo. Isso, entretanto, ocorre com a maioria das máquinas de oito bits quando rodam programas em BASIC e é difícil realizá-los em tempos menores.

Não há nenhum segredo em suas utilizações. Basta digitá-los e executá-los!

Se a execução demorar muito, faça alguma outra coisa enquanto espera. Por exemplo, vá lendo a análise e tentando entender o funcionamento dos programas.

DIGITAÇÃO

Estes programas possuem muitas instruções inseridas numa mesma linha e separadas por dois pontos (:). Tome muito cuidado para não pular algumas delas, pois isso pode alterar totalmente os efeitos produzidos.

Os programas são curtos. Em cinco minutos você digita qualquer um deles. Vale a pena tomar um pouco mais de cuidado durante a primeira digitação para não perder tempo procurando eventuais erros depois.

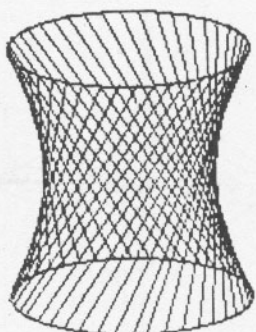
TWO-LINER 1 - CIRCLINE

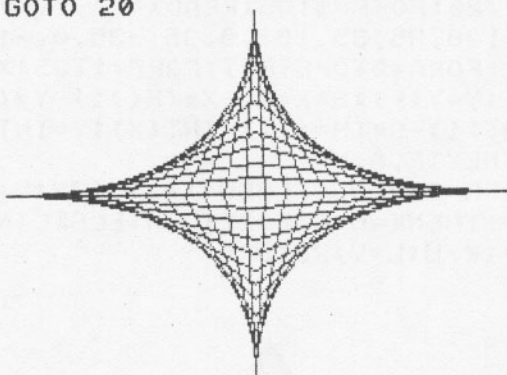
```
10 SCREEN 2
20 CC=128:LC=80:R=60:FORT=0TO359STEP5:G
G=2+14*RND(-TIME):X=T*3.141592#/180:C=C
C+R*COS(X):L=LC+R*SIN(X):LINE(CC,LC)-(C
,L),GG,B:NEXT:GOTO20
```

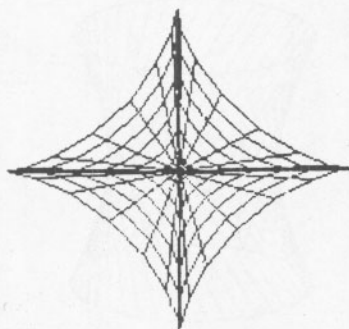
TWO-LINER 2 - CORES

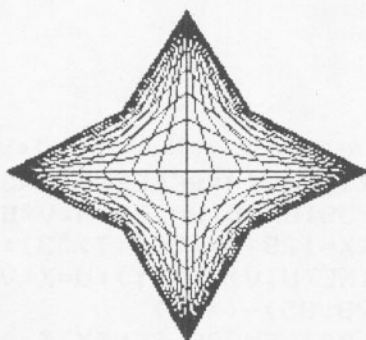
```
10 SCREEN2
20 FORF=0TO125STEP10:G=80-F*80/125:G1=2
+14*RND(-TIME):G2=2+14*RND(-TIME):G3=2+
14*RND(-TIME):G4=2+14*RND(-TIME):LINE(F
,80)-(125,G),G1,BF:LINE(125,G)-(250-F,8
0),G2,BF:LINE(F,80)-(125,160-G),G3,BF:L
INE(125,160-G)-(250-F,80),G4,BF:NEXT:GO
TO20
```

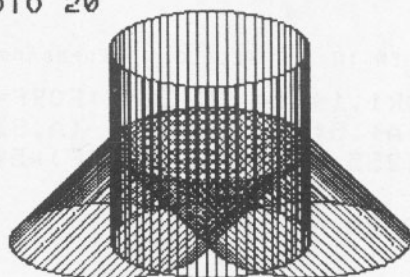
TWO-LINER 3 - CESTO

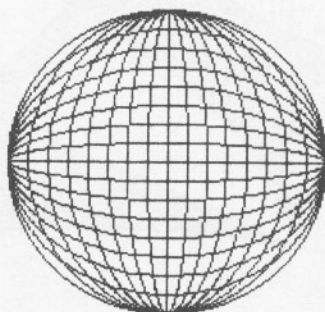
```
10 SCREEN2:P=3.1415926#:CIRCLE(128,30),
60,,,20/60:CIRCLE(128,160),60,,,20/60
:FORD=0TO2*PSTEP.15:E=D+P/2:X=128+60*CO
S(D):Y=30+20*SIN(D):S=128+60*COS(E):T=1
60+20*SIN(E):LINE(X,Y)-(S,T):NEXTD
20 GOTO 20
```

TWO-LINER 4 - ASTRÓIDE 1 (Rede)

```
10 SCREEN2:FORF=0TO125STEP10:G=80-F*80/
125:LINE(F,80)-(125,G):LINE(125,G)-(250
-F,80):LINE(F,80)-(125,160-G):LINE(125,
160-G)-(250-F,80):NEXT
20 GOTO 20
```

TWO-LINER 5 - ASTRÓIDE 2 (Teia)

```
10 COLOR1,14,14:SCREEN2:P=3.1415926#:FO
RA=10TO85STEP15:U=128+A:V=85:FORT=0TO2*
PSTEP.2:X=128+A*(COS(T)^3):Y=85+A*(SIN(
T)^3):LINE(U,V)-(X,Y):U=X:V=Y:IFA=85THE
NLINE(128,85)-(X,Y)
20 IFA=100THEN20ELSENEXTT,A:GOTO20
```

TWO-LINER 6 - ASTRÓIDE 3 (Estrela)

```
10 COLOR10,1:SCREEN2:P=3.1415926#:C=2*P
:I=P/20:FORF=1TO5:READX(F),Y(F):NEXT:DA
TA0,100,35,35,100,0,35,-35,0,-100:FORH=
0TO1:FORA=0TOPSTEPI:FORF=1TO5:X=X(F)*CO
S(A):Y=Y(F):S=X:X=-X*(H<>1)-Y*(H=1):Y=-
Y*(H<>1)-S*(H=1):X=INT(X):Y=INT(Y):GOSU
B20:NEXTF,A,H
20 IFH=2THEN20ELSEU=X*.9+128:V=Y*.9+85:
IFF=1THENK=U:L=V:RETURN:ELSELINE(K,L)-(
U,V):K=U:L=V:RETURN
```

TWO-LINER 7 - DRAWLIPSE

```
10 COLOR1,8,8::SCREEN2:FORA=ØTO6.28STEP
.1:PSET(128+50*COS(A),85+20*SIN(A)):DRA
W"nu50nf50nd50ng50":NEXTA:CIRCLE(128,34
),50,1,,,.4:CIRCLE(128,135),50,1,,,.4:C
IRCLE(128,85),50,1,,,.4:CIRCLE(77,135),
50,1,,,.4:CIRCLE(178,135),50,1,,,.4
20 GOTO 20
```

TWO-LINER 8 - ESFERA

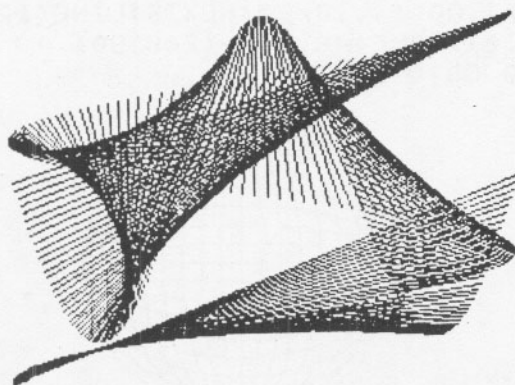
```
10 COLOR1,7,7:SCREEN2:FORB=80TO1STEP-10
:CIRCLE(128,80),80,1,,,80/B:CIRCLE(128,
80),80,1,,,B/80:NEXTB:LINE(128,160)-(12
8,0):LINE(48,80)-(208,80)
20 GOTO 20
```

TWO-LINER 9 - MONTES

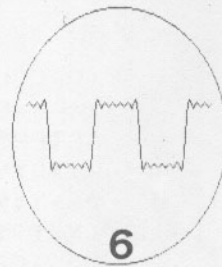
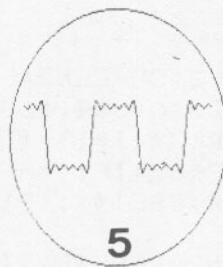
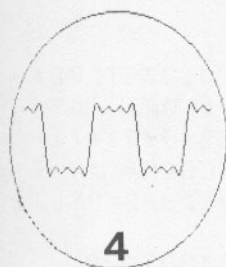
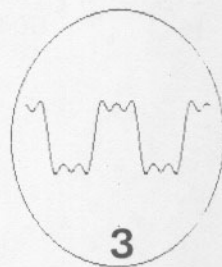
```
10 COLOR4,7,7:SCREEN2:FORX=0TO191:X1=X1
+255/191:P=X1*ATN(1)/45*360/255*3:Y=COS
(P)*130+128:LINE(X1,0)-(Y,X):NEXTX
20 GOTO 20
```

TWO-LINER 10 - TECIDO 1 (Enrugado)

```
10 COLOR1,14,14:SCREEN2:FORF=0TO25.2STE
P.05:A=A+.5:LINE(255,0)-(A,SIN(F)*50+90
):LINE(255,0)-(A-.5,SIN(F)*50+88),14:NE
XTF
20 GOTO 20
```

TWO-LINER 11 - TECIDO 2 (Torcido)

```
10 IFA=192THEN10ELSECOLOR4,7,5:SCREEN2:
FORA=0TO191:B=A*ATN(1)/90*360/255*4:C=C
+255/191:D=C*ATN(1)/90*360/255*3
20 E=SIN(B)*85+85:F=COS(D)*127+127:LINE
(C,E)-(F,A):NEXTA:GOTO 10
```

TWO-LINER 12 - FOURIER

```
1 COLOR15,12,12:PI=3.141592#:IFSTRIG(0)
()-1ANDP()0THEN1
2 P=P+1:SCREEN2:CIRCLE(128,85),84,1:PAI
NT(128,168),1:FORX=55TO200:Z=0:C=-X*(X=
55)-C*(X()55):FORM=1TOPSTEP2:Y=80/(M*PI
)*SIN(X*M*PI/36):Z=Z+Y:NEXT:Y=Z+85:L=-Y
*(X=55)-L*(X()55):LINE(C,L)-(X,Y),15:PS
ET(X,Y):C=X:L=Y:NEXT:P=P+1:GOTO1
```

Este TWO-LINER faz com que a tela se assemelhe ao visor de um osciloscópio.

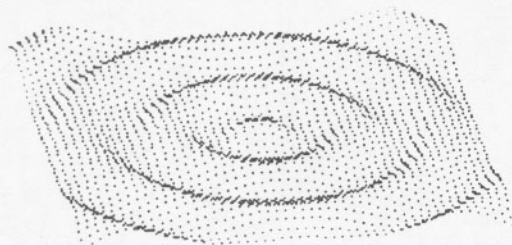
Ao ser executado, ele permite visualizar a forma da onda resultante da soma dos harmônicos ímpares de uma frequência qualquer. A primeira imagem é a do primeiro harmônico; a segunda é a da soma do primeiro com o terceiro harmônico; a terceira é a soma do primeiro com o terceiro e com o quinto harmônico; e assim por diante...

Para mudar de uma imagem para outra digite a barra de espaços.

TWO-LINER 13 - COGUMELO 1

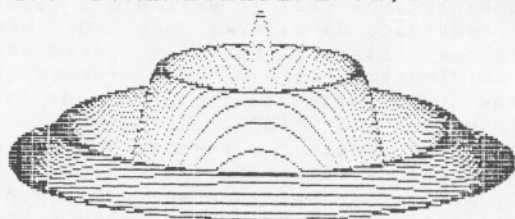
```
10 SCREEN2
20 FORZ=0TO10STEP.1:FORX=0TO10STEP.1:Y=
-(COS(3*SQR((X-5)*(X-5)+(Z-5)*(Z-5))))*
5+30:PSET(X*24+Z,191-Y-Z*10):NEXTX,Z:GO
TO20
```

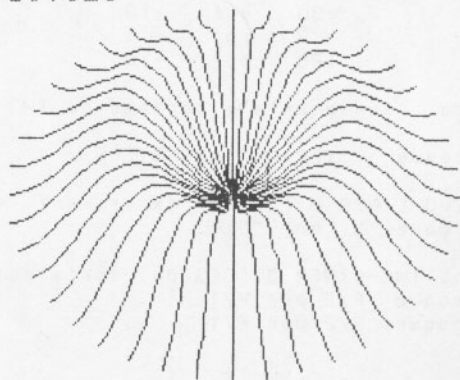
TWO-LINER 14 - COGUMELO 2

```
10 SCREEN2:FORZ=0TO10STEP.2:FORX=0TO10S
TEP.2:Y=-10*COS(3*SQR((X-5)*(X-5)+(Z-5)
*(Z-5)))/2+50:PSET(X*20+20+Z*3,Y+Z*10):
NEXTX,Z
70 GOTO 70
```

TWO-LINER 15 - COGUMELO 3

```
10 COLOR1,3,3:SCREEN2:FORB=0TO127STEP2:
X4=B*B:M=-128:A=SQR(16384-X4):FORI=-ATO
ASTEP5:R=SQR(X4+I*I)/128:F=COS(9*R)*(1-
R)*2:Y=I/5+F*32:IFY<=MTHEN20ELSEM=Y:Y=1
28+Y:X=128+B:PSET(X,191-Y):X=128-B:PSET
(X,191-Y)
20 IFP=1THEN20ELSENEXTI,B:P=1:GOTO20
```

TWO-LINER 16 - COGUMELO 4

```
10 COLOR1,5,5:SCREEN2:FORB=0TO127:X4=B*
B:M=-128:A=SQR(16384-X4):FORI=-ATOASTEP
3:R=SQR(X4+I*I)/128:F=COS(16*R)*(1-R)*2
:Y=I/5+F*32:IFY<=MTHEN20ELSEM=Y:Y=128+Y
:X=128+B:PSET(X,191-Y):X=128-B:PSET(X,1
91-Y)
20 IFP=1THEN20ELSENEXTI,B:P=1:GOTO20
```

TWO-LINER 17 - ANÊMONA

```
10 COLOR1,3,3:SCREEN2:P=3.1415926#:Q=2*
P:R=P/20:S=P/10:T=60:U=150/Q:FORB=0TOQS
TEPR:C=T*P/180:C=(B>P)*C-(B<=P)*C:FORH=
0TOQSTEPS:Y=COS(H)*42:X=COS(B)*H*U:Z=-(
SIN(B)=1)*H*U-(SIN(B)<>1)*TAN(B)*X:Z=AB
S(Z):Z=-1E-03*(Z=0)-Z*(Z<>0):J=SQR(Z*Z+
Y*Y):A=ATN(Y/Z)+C
20 IFF=1THEN20ELSEY=J*SIN(A):X=INT(X):Y
=INT(Y):XP=X*.7+128:YP=Y*.7+85:XO=-XO*(
H<>0)-XP*(H=0):YO=-YO*(H<>0)-XP*(H=0):L
INE(XO,YO)-(XP,YP):XO=XP:YO=YP:NEXTH,B:
F=1:GOTO20
```

ANÁLISE

Não iremos detalhar o funcionamento de cada um dos programas. Todos eles envolvem um certo conhecimento de matemática elementar e, na verdade, são bastante simples.

Se você tiver uma boa e sólida formação matemática ao nível do segundo grau e conhecer bem o BASIC MSX, poderá compreender sozinho os TWO-LINERS. Caso contrário, por mais que tentássemos, nas poucas linhas de que dispomos, não conseguiriamos fazê-lo entender a maioria deles!

O resultado da maioria dos TWO-LINERS apresentados são figuras interpretadas pelo nosso cérebro como superfícies bidimensionais. Há duas formas de se gerar esses efeitos. Uma é artificial, fazendo de fato um desenho num plano (os ASTRÓIDES, o CESTO, a ESFERA, o DRAWLIPSE, os MONTES, etc...) e nesse caso a execução é rápida, pois poucos cálculos são executados. A outra forma é projetando, através de sua equação, uma superfície realmente bidimensional no plano da tela. Isso é bem mais trabalhoso e em geral demora muito mais tempo (os COGUMELOS, a ANÊMONA, etc...).

Outros TWO-LINERS geram figuras no próprio plano da tela e são normalmente mais simples (CIRCLINE, FOURIER).

Para tentar compreender um TWO-LINER, é conveniente decompô-lo em várias linhas, de modo a facilitar a análise individual de cada instrução. Tente fazer isso com alguns deles, por exemplo, com a ESFERA!

Outra importante técnica (empírica!) para identificar as partes principais de um TWO-LINER é ir alterando um a um seus valores numéricos ou suas funções. Experimete fazer as seguintes alterações, uma de cada vez, no TWO-LINER 14 (COGUMELO 2).

* Troque a função COS pela função SIN.
* Troque:

```
Z*10   por  Z*15   ;
 +50   por   +10   ;
  .2   por    .1   e
 +20   por   +10   .
```

Agora, tente esta alteração no TWO-LINER 4 (REDE):

* Troque STEP10 por STEP5 .

No TWO-LINER 5 (TEIA), experimente esta:

* Troque ^3 por ^5 .

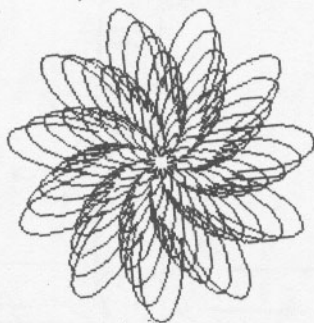
E, no TWO-LINER 3 (CESTO), tente estas:

* Troque P/2 por P/1.5 .
* Troque P/2 por P/1 .

Agora é a sua vez de propor (e realizar!) alterações. Comece pelo primeiro programa e siga até o último.

Depois, terá chegado a hora de você fazer seus próprios TWO-LINERS. Inicialmente, faça o programa em várias linhas. A seguir, compacte-o em apenas duas. Isso, ao contrário do que parece, não é muito simples! Para certificar-se disso, digite e execute algumas vezes o THREE-LINER Pétalas. Depois, tente transformá-lo num TWO-LINER! Se você coinseguir, avise-nos! Isso é extremamente difícil!!!

THREE-LINER 1 - PÉTALAS

```
10 INPUT"Quantas petalas";P:INPUT"Com r
otacao (s/n)";A$:PI=3.141592#:Q=2*PI:CO
LOR1,8,8:SCREEN2:N1=5:N2=4:FORV=1TO10:I
=V*PI/30*(A$="s"):M1=N1*V:M2=N2*V:FORF=
0TOQSTEPQ/180:A=F-I:S=ABS(SIN(F*P/2)):R
=S*M1+M2:X=COS(A)*R:Y=SIN(A)*R:GOSUB30:
X=INT(X):Y=INT(Y)
20 IFIP=1THEN20ELSENEXTF:GOSUB30:NEXTV:
IP=1:GOTO 20
30 XP=X*.9+128:YP=191-(Y*.9+85):IFF=0TH
ENX1=X:Y1=Y:XO=XP:YO=YP:RETURN20:ELSELI
NE(XO,YO)-(XP,YP):XO=XP:YO=YP:RETURN
```

Este programa pede a introdução de duas informações: o número de pétalas e se elas devem ser rotacio-

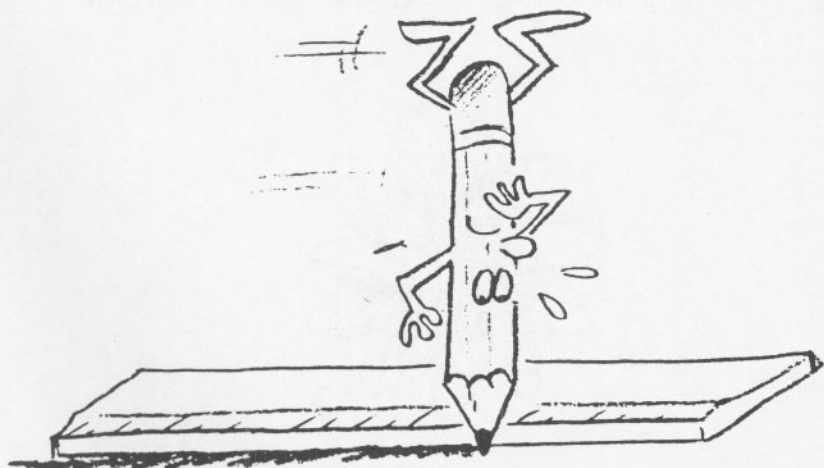
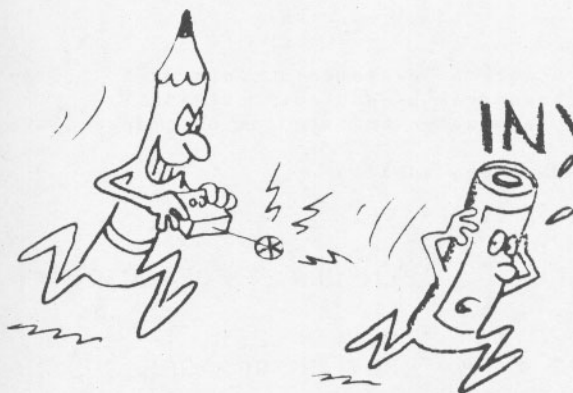
nadas ou não.

Experimente as seguintes combinações:

NÚMERO DE PÉTALAS?	ROTAÇÃO?
5	não
8	sim
9	não
11	sim

Um caminho mais fácil para começar a fazer TWO-LINERS, principalmente se você é bastante egocêntrico, é começar com programas do tipo EGO . Experimente!

TWO-LINER 18 - EGO

```
10 SCREEN2:PSET(90,80):DRAW"f1d10g1bm+1
,-11r2f1d3g1l2r2d1f1d2f1r1bm+2,-2r1u1h1
g1d2f1r1bm+1,-4f1d3u3e1f1d3r1bm+1,-4r1f
1d3r1l1u2l2d2r2bm+2,-9d8f1bm-3,-6r4bm+1
,+2g1d2f1e1u2h1bm+9,-3h1l2g1d2f1r2f1d2g
1l2h1bm+9,-6d6f1r2e1u6h1l2g1bm-3,+7r0bm
+9,0r0"
20 GOTO 20
```

SAÍDA INVISÍVEL

INSTRUÇÕES

Você se encontra num lugar repleto de armadilhas. Só existe uma saída, que você não vê, e sua missão é encontrá-la o mais rapidamente possível.

Para que a saída seja atingida, você deve coletar as quatro pilhas que estão espalhadas em pontos aleatórios da tela. Elas servirão para alimentar o seu receptor e, quando a última for apanhada, você passará a receber sinais que são emitidos constantemente por um microtransmissor instalado na saída. Esses sinais são decodificados por seu receptor que informa a direção a seguir. Na parte inferior da tela aparecem as palavras NORTE ou SUL e LESTE ou OESTE. Por exemplo, se surgirem as informações SUL e OESTE, significa que a saída está abaixo e à esquerda da posição ocupada por você. Caso essas palavras não apareçam, isto quer dizer que você está na direção correta.

Com relação ao contador de tempo, ele funciona como um taxímetro. Se você ficar parado numa determinada posição, ele não pára. Caso você se mova, ele se adianta em dez unidades para cada posição avançada. Convém, portanto, planejar o caminho mais curto e mais rápido para alcançar as pilhas e depois a saída.

Para se movimentar utilize as teclas de setas. Cuidado para não esbarrar em qualquer uma das armadilhas enquanto estiver se movendo pois, nesse caso, você fica preso e o jogo termina.

Quando você alcançar a saída, o contador de tempo pára e o programa checa se o recorde foi batido. Desta forma, o jogo acaba e a pressão da tecla "R" o reinicia.

DIGITAÇÃO

Digite o programa exatamente conforme a listagem. Você não deve encontrar grandes dificuldades.

Confira-o e grave-o em fita com o comando CSAVE "SAINV".

Comande RUN e boa sorte!

```
1 REM SAIDA INVISIVEL
10 GOSUB 740
20 MAXFILES=1 : OPEN"GRP:" FOR OUTPUT A
S #1
30 DIM M(28,16) : SCREEN 2,0
40 RE=9999 : D$="R7D7L7U7BF2R3D3L3U3" :
 A$="R6D6L5U4R3D2L1"
50 SPRITE$(1)=CHR$(0)+CHR$(24)+CHR$(60)
+CHR$(60)+CHR$(60)+CHR$(60)+CHR$(60)+CH
R$(0)
60 SPRITE$(2)=CHR$(60)+CHR$(66)+CHR$(16
5)+CHR$(129)+CHR$(165)+CHR$(153)+CHR$(6
6)+CHR$(60)
70 SCREEN 2 : COLOR 15,1,1 : CLS
80 TE=0 : PI=0 : TT=0 : W=0
90 FOR I=8 TO 240 STEP 8
100 PSET(I,24),0 : COLOR 5 : DRAW D$
110 PSET(I,160),0 : COLOR 5 : DRAW D$
120 NEXT I
130 FOR I=32 TO 152 STEP 8
140 PSET(8,I),0 : COLOR 5 : DRAW D$
150 PSET(240,I),0 : COLOR 5 : DRAW D$
160 NEXT I
170 PSET(72,0),0: COLOR 8 : PRINT #1,"S
AIDA INVISIVEL"
180 GOSUB 620
190 GOSUB 610
200 Q=INT(RND(-TIME)*1000)+1
210 FOR K=1 TO 65
220 X=INT(RND(Q)*27)+2 : I=X*8+8
230 Y=INT(RND(Q)*16)+1 : J=Y*8+24
240 IF M(X,Y) <> 0 THEN 220
250 IF K < 61 THEN M(X,Y)=5 ELSE GOTO 2
80
```

```
260 PSET(I,J),0 : COLOR 8 : DRAW A$
270 GOTO 320
280 IF K < 65 THEN M(X,Y)= K-60  ELSE G
OTO 310
290 PUT SPRITE M(X,Y),(I,J),11,1
300 GOTO 320
310 XS=X : YS=Y : M(X,Y)=6
320 NEXT K
330 X=1 : Y=1 : I=16 : J=32
340 PUT SPRITE 0,(I,J),7,2
350 IF INKEY$="" THEN 350
360 T=STICK(0)
370 IF T=1 OR T=3 OR T=5 OR T=7 THEN 41
0
380 W=W+1
390 IF W=5 THEN TE=TE+1 : GOSUB 610 : W
=0
400 GOTO 360
410 TE=TE+10 : GOSUB 610
420 IF T=1 AND Y-1 > 0 THEN Y=Y-1 : J=Y
*8+24 : GOTO 460
430 IF T=3 AND X+1 <= 28 THEN X=X+1 : I
=X*8+8 : GOTO 460
440 IF T=5 AND Y+1 <= 16 THEN Y=Y+1 : J
=Y*8+24 : GOTO 460
450 IF T=7 AND X-1 > 0 THEN X=X-1 : I=X
*8+8
460 PUT SPRITE 0,(I,J),7,2
470 IF M(X,Y)=5 THEN PLAY"T255L64CDEFGC
DEFGCDEFGCDEFGCDEFG" : GOTO 730
480 IF M(X,Y)>0 AND M(X,Y)<5 THEN PLAY"
T255L64BEB" : GOSUB 630
490 IF PI=4 THEN GOSUB 650
500 IF TT=1 THEN PLAY"T255L64CDEEFEFCAC
A" : GOTO 520
510 GOTO 360
520 IF RE > TE THEN RE=TE : GOSUB 620
530 LINE (88,88)-(175,95),1,BF : PSET(8
8,88),0 : COLOR 15 : PRINT #1,"FIM DO J
OGO"
```

```
540 FOR X=1 TO 28
550 FOR Y=1 TO 16
560 M(X,Y)=0
570 NEXT Y : NEXT X
580 PSET(0,184),0 : COLOR 15 : PRINT #1
,"Aperte a tecla R para recomecar."
590 IF INKEY$ <> "R" THEN 590
600 GOTO 70
610 LINE(216,16)-(247,23),1,BF :  PSET(
168,16),0 : COLOR 2 : PRINT #1,USING"TE
MPO:####";TE : RETURN
620 LINE(72,16)-(103,23),1,BF : PSET(8,
16),0 : COLOR 2 : PRINT #1,USING"RECORD
E:####";RE : RETURN
630 PI=PI+1 : PUT SPRITE M(X,Y),(I,Y),0
,1 : M(X,Y)=0 : IF PI=4 THEN PLAY"T255L
64CAC"
640 RETURN
650 IF X < XS THEN LINE (176,168)-(215,
175),1,BF : PSET(176,168) : COLOR 2 : P
RINT #1,"LESTE"
660 IF X = XS THEN LINE (176,168)-(215,
175),1,BF
670 IF X > XS THEN LINE (176,168)-(215,
175),1,BF : PSET(176,168) : COLOR 2 : P
RINT #1,"OESTE"
680 IF Y > YS THEN LINE (40,168)-(79,17
5),1,BF : PSET(40,168) : COLOR 2 : PRIN
T #1,"NORTE"
690 IF Y = YS THEN LINE (40,168)-(79,17
5),1,BF
700 IF Y < YS THEN LINE (40,168)-(79,17
5),1,BF : PSET(40,168) : COLOR 2 : PRIN
T #1,"S U L"
710 IF M(X,Y)=6 THEN TT=1
720 RETURN
730 TE=9999 : GOSUB 610 : GOTO 530
740 CLS : COLOR 4,1 : KEYOFF
750 SCREEN 0
760 LOCATE 12,5
```

```
770 PRINT"SAIDA INVISIVEL"
780 LOCATE 5,17
790 PRINT"Aperte uma letra para comecar
...."
800 IF INKEY$="" THEN GOTO 800
810 RETURN
```

ANÁLISE

A idéia fundamental do programa é mapear a tela associando a cada uma de suas posições um elemento de uma matriz de 28 colunas por 16 linhas (M(28,16)). A posição da tela que estiver livre tem, como correspondência na matriz, um elemento igual a zero.A posição que contiver uma armadilha assume o valor igual a 5; as que contiverem as pilhas recebem os valores 1 a 4 e a que contiver a saída terá valor igual a 6. Isso facilita as tomadas de decisões lógicas do programa. Além disso as coordenadas da saída são armazenadas nas variáveis XS e YS.

A linha 10 do programa o remete a uma sub-rotina de apresentação (linhas 740 a 830) cuja função é tornar aleatório o instante em que a execução do programa propriamente dito começa. Assim, a variável TIME, que será utilizada na linha 200, realmente assume um valor qualquer. Se isso não fosse feito, o tempo que decorreria desde o instante em que o comando RUN fosse acionado até a execução da linha 200 seria sempre o mesmo, e o padrão da tela a ser construído seria sempre repetido. O jogo perderia a "graça" pois,após rodá-lo algumas vezes, você já saberia de cor onde está a saída.

A linha 20 abre um arquivo na tela de alta resolução para que possamos escrever nela posteriormente.

A linha 30 dimensiona a matriz de posições da tela e seleciona a tela de alta resolução apenas para que os SPRITES possam ser definidos nas linhas seguintes.

A linha 40 atribui o valor inicial ao recorde (variável RE) e define as cadeias de caracteres que serão posteriormente utilizadas como argumento da instrução DRAW.

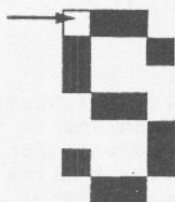
As linhas 50 e 60 definem respectivamente os SPRITES correspondentes ao desenho de uma pilha e de uma cara sorridente que representará o jogador no programa.

A linha 80 zera o contador de tempo (variável TE), o contador de pilhas já coletadas (variável PI), a "flag" que testa se a saída foi atingida (variável TT) e a variável retardadora do avanço de tempo enquanto, durante o jogo,você estiver parado em alguma posição (variável W).

Os laços definidos entre as linhas 90 e 160 têm, por finalidade, desenhar a moldura.

A linha 170, utilizando o arquivo aberto na li-

nha 20, escreve o título do jogo na tela de alta resolução. A função da instrução PSET empregada nessa linha é indicar a posição do vértice superior esquerdo do campo referente ao primeiro caractere a ser impresso.

Último PSET antes do primeiro PRINT # 1 "S...

As linhas 180 e 190 remetem a execução às subrotinas que imprimem respectivemente o RECORDE e o TEMPO. Além da técnica já usada na linha 170 (uso do PSET), elas pintam previamente com a cor de fundo (preto), a região onde os valores são impressos.

A linha 200 sorteia, aleatoriamente, um valor entre 1 e 1000 que servirá de argumento para a função RND utilizada na "construção" da tela.

As linhas entre 210 e 320 "espalham" aleatoriamente 60 armadilhas, 4 pilhas e uma saída na tela. No laço definido por essas linhas, são também atribuídos os valores 5, 1 a 4, e 6 para os elementos da matriz de posições (M(X,Y)) que conterão as armadilhas, as pilhas e a saída, respectivamente. Note que os valores atribuídos aos elementos da referida matriz correspondentes às pilhas são também os números da camada (1 a 4) em que o SPRITE relativo à pilha é colocado (linhas 280 e 290). Essa técnica é fundamental para que, posteriormente, o programa possa "reconhecer" a camada em que se encontra a pilha alcançada pelo jogador e possa apagá-la na tela.

Nas linhas 220 e 230 são calculados os valores das variáveis I e J que se correspondem com as coordenadas X e Y, respectivamente. A diferença entre o par (X,Y) e o par (I,J) é que o primeiro define uma posição da matriz M(X,Y) e corresponde na tela de alta resolução a 8x8 pontos, enquanto que o segundo define a posição do ponto de alta resolução da tela correspondente ao vértice superior esquerdo do par (X,Y).

A linha 330 atualiza as coordenadas em (1,1) na matriz e (16,32) na tela. Elas são relativas ao canto superior esquerdo da tela. A linha 340 coloca você (SPRITE$(2)) na camada 0 e nas coordenadas atuais, e a linha 350 "espera" que uma tecla seja pressionada para que o jogo comece.

A linha 360 armazena na variável T o estado das teclas de seta pressionadas. Se nenhuma seta for digitada, a variável W é incrementada, e quando atinge o valor 5, o tempo sofre o acréscimo de uma unidade (linhas 380 a 400). Se, por outro lado, qualquer uma das setas for pressionada, o programa é desviado para a linha 410, que incrementa o contador de tempo em 10 unidades. Entre as linhas 420 e 450, testa-se se o movimento correspondente à seta digitada é ou não possível. Caso seja, as coordenadas da matriz (X,Y) e da tela (I,J) são atualizadas e você é reposicionado pela linha 460.

A linha 470 verifica se a nova posição corresponde a uma armadilha. Em caso positivo o programa é desviado para a linha 730 e o jogo termina.

A linha 480 verifica se você alcançou uma pilha. Em caso positivo, será executada a sub-rotina 630, que incrementa a variável PI (contadora do número de pilhas coletadas) e "apaga" a pilha recolhida, reconhecendo o número da camada em que ela (SPRITE) se encontra, através do valor da matriz M(X,Y). A seguir, esse elemento é "zerado", para se tornar uma posição livre para movimento.

No retorno da sub-rotina descrita acima, será verificado pela linha 490 se todas as pilhas já foram alcançadas. Em caso afirmativo, será executada a sub-rotina 650 que "liga" o seu receptor, imprimindo as direções a serem seguidas, e verifica se a saída foi atingida.

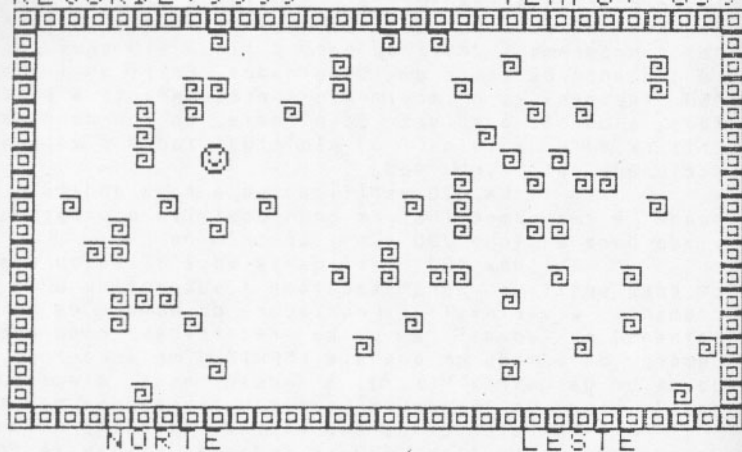
Quando a saída for alcançada (TT=1), a linha 500 toca algumas notas musicais e desvia o programa para a linha 520 que, por sua vez, verifica se o recorde foi batido.

As linhas 530 a 600 finalizam o jogo. Entre elas há um laço (linhas 540 a 570) que "limpa" todas as posições da matriz M(X,Y) para que o jogo possa recomeçar.

Agora que você já sabe como o programa funciona em linhas gerais, experimente fazer algumas modificações.

Por exemplo, tente definir novas figuras para construir a moldura e as armadilhas usando o comando DRAW.

Tente também definir outros SPRITEs para as pilhas e para a cara que representa você no jogo.

Outra modificação que você pode tentar fazer para tornar o jogo mais difícil é eliminar as indicações de direção (NORTE, SUL, LESTE e OESTE). Nesse caso, as pilhas devem ser colhidas apenas para que a saída se torne acessível e a estratégia do jogo muda: você deve evitar passar mais do que uma vez num mesmo lugar.

Tela do saída invisível após colher as pilhas

POUSO

INSTRUÇÕES

Você é o piloto de um caça supersônico que está voltando à base, seriamente avariado após uma vitoriosa batalha.

No caminho de volta, o caça acabou ficando sem combustível e começou a perder altitude rapidamente.

Sua única salvação é fazer um POUSO FORÇADO numa pista inimiga que está abandonada!

A cada segundo que passa seu caça supersônico se aproxima mais e mais da pista, que está cheia de obstáculos, tornando quase impossível qualquer tentativa de pouso sem que haja explosão do seu avião.

Para conseguir pousar você terá que desobstruir a pista, lançando bombas sobre os obstáculos, até que ela fique totalmente limpa. Se você conseguir acertar os obstáculos e deixar a pista limpa, sua aterrissagem será tranquila, caso contrário...

Para lançar as bombas basta apertar a barra de espaços, mas devido às avarias sofridas durante a batalha, seu caça só consegue lançar uma bomba de cada vez.

Faça uma boa mira e bom pouso!!!

DIGITAÇÃO

Para começar a digitação do programa digite o comando AUTO 1000,10 . Com isso o micro lhe fornecerá automaticamente os números das linhas, iniciando em mil e indo de dez em dez. Você notará que com essa numeração a listagem de seu programa ficará mais estética.

Preste muita atenção ao digitar cada linha, pois uma vírgula a menos ou a mais será o bastante para que o programa não funcione.

Ao terminar cada linha, verifique se ela está correta e se não estiver, volte com o cursor até o erro e corrija-o. Quando tiver certeza que a linha está totalmente correta, digite RETURN.

Nesse instante o micro lhe fornecerá o número da próxima linha que deverá ser digitada.

Após ter introduzido todo o programa, digite CONTROL+STOP para interromper a numeração automática e grave o programa em fita com: CSAVE"POUSO" + RETURN.

Para iniciar o jogo, basta comandar RUN+RETURN (ou a tecla F5).

A única observação a ser feita está na linha de número 2350, onde o caractere, semelhante a uma árvore, que está após a primeira aspa que aparece, é obtido pressionando-se as teclas LGRA + [;] .

```
1000 '------------------------------------
1010 '     jogo : Pouso forcado
1020 '
1030 '     Rubens      nov / 85
1040 '------------------------------------
1050 WIDTH 40
1060 DEFINT A-Z
1070 MAXFILES = 1
1080 X = RND(-TIME)
1090 '
1100 '    INSTRUCOES
1110 '
1120 KEY OFF
1130 SCREEN 0,,0
1140 COLOR 1,7,7
1150 CLS
1160 LOCATE 8,2,1
1170 PRINT ">> POUSO FORCADO <<"
1180 LOCATE 0,6,0
1190 PRINT "Voce tem que fazer um ";
1200 PRINT "pouso forcado num"
1210 PRINT "campo  inimigo  onde ";
1220 PRINT " foram  colocados"
1230 PRINT "varios obstaculos para";
1240 PRINT " impedir o pouso."
1250 LOCATE 0,11,0
1260 PRINT "Para  desobstruir  a ";
1270 PRINT " pista  voce  deve"
```

```
1280 PRINT "lancar bombas sobre";
1290 PRINT " os obstaculos, ate"
1300 PRINT "que  a  pista  fique";
1310 PRINT " totalmente limpa."
1320 LOCATE 0,17,0
1330 PRINT "Use a barra de espaco";
1340 PRINT " para  bombardear."
1350 LOCATE 15,22,0
1360 PRINT "TECLE QUALQUER TECLA ";
1370 X$ = INPUT$(1)
1380 '
1390 '   ROTINA PRINCIPAL
1400 '
1410 SCREEN 2,0
1420 OPEN "GRP:" FOR OUTPUT AS #1
1430 GOSUB 1670
1440 GOSUB 1860
1450 GOSUB 2290
1460 GOSUB 2420
1470 COLOR 1
1480 PRESET (60,20)
1490 IF IC$="s" THEN GOSUB 1620 ELSE GO
SUB 1570
1500 IF INKEY$ <> "" THEN GOTO 1500
1510 PRESET (60,60)
1520 PRINT #1,"TECLE QUALQUER TECLA";
1530 X$ = INPUT$(1)
1540 LINE (60,20)-(220,28),7,BF
1550 LINE (60,60)-(220,68),7,BF
1560 GOTO 1450
1570 '
1580 '   ATERRISSOU
1590 '
1600 PRINT #1,"POUSO  PERFEITO  !!!"
1610 RETURN
1620 '
1630 '   COLIDIU
1640 '
1650 PRINT #1,"SEU AVIAO EXPLODIU !"
1660 RETURN
```

```
1670 '
1680 '    LER SPRITES
1690 '
1700 RESTORE
1710 FOR I = 1 TO 3
1720     Y$ = ""
1730     FOR J = 1 TO 8
1740         READ X$
1750         Y$=Y$+CHR$(VAL("&H"+X$))
1760     NEXT J
1770     SPRITE$(I)=Y$
1780 NEXT I
1790 RETURN
1800 '
1810 '    DEFINICAO DOS SPRITES
1820 '
1830 DATA 00,40,60,70,78,FF,7F,00
1840 DATA 00,00,00,1C,7E,FF,FF,00
1850 DATA 00,38,10,38,7C,38,10,00
1860 '
1870 '    PAISAGEM
1880 '
1890 LINE (0,102)-(255,191),10,BF
1900 FOR I = 105 TO 190 STEP 5
1910     X1 = 256 * RND(1)
1920     X2 = 256 * RND(1)
1930     LINE (X1,I)-(X2,I),1
1940 NEXT I
1950 PSET (0,100),12
1960 DRAW "R4U2R4D1R5D1R4U1R2U1R4U1R3"
1970 DRAW "U1R4U1R2U2R5D1R2D1R2D1R5U2"
1980 DRAW "R5U1R4U1R5U1R5U1R4U1R2E5U2"
1990 DRAW "R1U3R2E4R2F5D4R3F3R8F2R7U1"
2000 DRAW "R2U1R9F2R1E3R1E4R4U3R3U2R5"
2010 DRAW "E3R3U2R2U1R3D1R2F4D3R3F4D3"
2020 DRAW "R4F3R7D1R7D1R9E2R4F2R2E3R2"
2030 DRAW "F3R4F2R4D1R8D1R6D1R7D2R5"
2040 LINE  (0,102)-(255,102),12
2050 PAINT (0,101),12
2060 PSET (56,101),2
```

```
2070 DRAW "R4U2R4D1R5U1R4D1R2U1R4U1R3"
2080 DRAW "U1R4U1R2U2R5D1R2D1R2D1R5D2"
2090 DRAW "F2"
2100 LINE (0,102)-(255,102),2
2110 PAINT (80,100),2
2120 PSET (190,101),2
2130 DRAW "R4U2R4D1R5U1R4D1R2U1R4U1R3"
2140 DRAW "U1R4U1R2U2R5D1R2D1R2D1R5D2"
2150 DRAW "F2"
2160 PAINT (200,101),2
2170 CIRCLE (25,20),8,11
2180 PAINT (25,20),11
2190 LINE (128,5)-(150,6),15,BF
2200 LINE (124,7)-(158,8),15,BF
2210 LINE (131,9)-(147,10),15,BF
2220 LINE (180,45)-(200,46),14,BF
2230 LINE (175,47)-(204,48),14,BF
2240 LINE (180,49)-(202,50),14,BF
2250 PSET (30,160),14
2260 DRAW "R150E20L150G20"
2270 PAINT (40,159),14
2280 RETURN
2290 '
2300 '   OBSTACULOS
2310 '
2320 LINE (58,143)-(180,152),14,BF
2330 AV$ = ""
2340 FOR I = 1 TO 8
2350    IF RND(1)> .5 THEN AV$ = AV$ +
"♣" ELSE AV$ = AV$ + "  "
2360 NEXT I
2370 IF AV$=STRING$(16,32) THEN GOTO 23
30
2380 COLOR 13
2390 PSET (60,145),14
2400 PRINT #1,AV$
2410 RETURN
2420 '
2430 '   ROTINA DO JOGO
2440 '
```

```
2450 YA=10
2460 XA=0
2470 TB$="N"
2480 ON STRIG GOSUB 2960
2490 STRIG(0) ON
2500 XA=XA+5 : IF XA > 240 THEN XA=0
2510 IF XA MOD 10 = 0 THEN YA=YA+1
2520 IF YA>127 AND XA=0 THEN GOTO 2600
2530 PUT SPRITE 15,(XA,YA),4,1
2540 PUT SPRITE 16,(XA+8,YA),4,2
2550 XB=XB+4 : IF XB > 248 THEN XB = 0
2560 YB=YB+5
2570 IF TB$ = "s" THEN PUT SPRITE 17,(X
B,YB),1,3
2580 IF YB>140 AND TB$="s" THEN GOSUB 2
880
2590 GOTO 2500
2600 '
2610 '   ATERRISSOU
2620 '
2630 STRIG(0) OFF
2640 IC$ = "s"
2650 FOR XA = 0 TO 35
2660      PUT SPRITE 15,(XA,140),4,1
2670      PUT SPRITE 16,(XA+8,140),4,2
2680 NEXT XA
2690 FOR XA = 36 TO 170
2700      PUT SPRITE 15,(XA,140),4,1
2710      PUT SPRITE 16,(XA+8,140),4,2
2720      IF POINT (XA+16,148) <> 14 THE
N GOTO 2760
2730 NEXT XA
2740 IC$ = "n"
2750 RETURN
2760 '
2770 '   EXPLOSAO DO AVIAO
2780 '
2790 FOR I = 1 TO 3
2800    FOR J = 2 TO 14
2810      PUT SPRITE 15,(XA,140),J,1
```

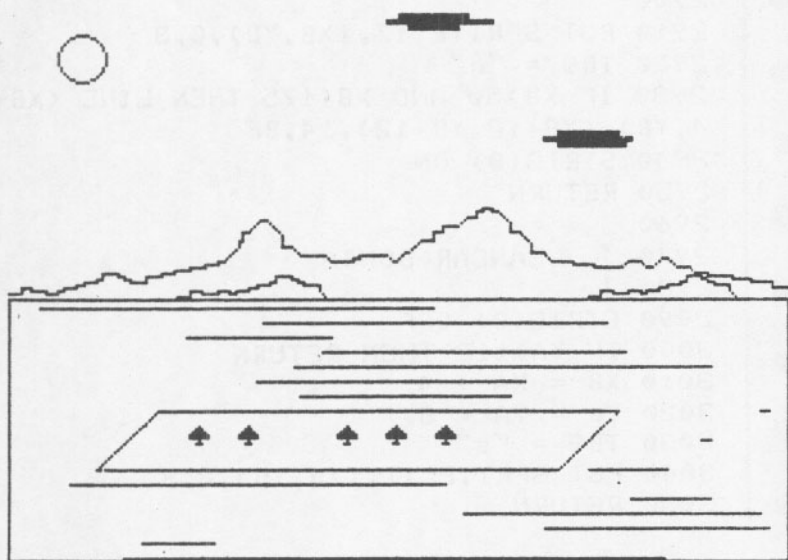
```
2820       PUT SPRITE 16,(XA+8,140),J,2
2830       FOR K = 1 TO 20
2840       NEXT K
2850     NEXT J
2860 NEXT I
2870 RETURN
2880 '
2890 '   EXPLOSAO DA BOMBA
2900 '
2910 PUT SPRITE 17,(XB,YB),0,3
2920 TB$ = "n"
2930 IF XB>50 AND XB<175 THEN LINE (XB-
4,YB)-(XB+12,YB+12),14,BF
2940 STRIG(0) ON
2950 RETURN
2960 '
2970 '   LANCAR BOMBA
2980 '
2990 STRIG(0) OFF
3000 IF YA>112 THEN RETURN
3010 XB = XA + 4
3020 YB = YA + 8
3030 TB$ = "s"
3040 PUT SPRITE 17,(XB,YB),8,3
3050 RETURN
```

ANÁLISE

Se você quiser fazer algumas modificações, aqui vão algumas sugestões:

Caso você queira modificar a paisagem, esta é definida desde a linha 1860 até a linha 2280. Tome cuidado com a cor da pista, que deverá ser necessariamente cinza (código 14) e não pode ser desenhada em outro local.

O desenho do avião e da bomba estão definidos nas linhas 1830, 1840 e 1850.

Se você quiser colocar som no programa, por exemplo, o de explosão da bomba, pode colocá-lo entre as linhas 2880 e 2950. Outra sugestão é a de colocar música para quando o pouso for perfeito ou outra música para o impacto do avião ou, ainda, o som do motor do avião enquanto ele sobrevoa a pista. Veja o último capítulo (TWO-LINERS SONOROS) para se inspirar.

INTRODUÇÃO

Estamos no ano de 3003 onde os esportes não exigem muita força, mas sim reflexos rápidos e muita velocidade.

Neste ano, na OLIMPIADA MSX, ocorrerá a mais almejada prova de todos os tempos: o CHIPTRON!

Essa prova se realiza numa arena circular onde dois TRONS devem dirigir suas motos sem se chocarem com as bordas, obstáculos ou rastros deixados pelas motos.

O objetivo de cada TRON é fazer com que seu adversário colida.

Cuidado com as bordas! Em algumas partes existem escorregadores que são passagens diretas para a morte.

As motos, ao partirem, acionam um contador de tempo que ao final será somado ao placar do vencedor.

Para controlar as motos, use as teclas A,W,D,X (TRON esquerdo) e 2,4,6,8 (TRON direito). As teclas de comando são apenas para o controle direcional e não para a movimentação.

Para iniciar o jogo você deve acionar a tecla CAPS LOCK antes de comandar RUN + RETURN ou F5.

DIGITAÇÃO

As linhas de 180 a 250, de 260 a 330, de 90 a 120, de 400 a 530, e 370 e 380 devem ser digitadas como foi explicado no capítulo 1, para reduzir o tempo de digitação.

As linhas 140 e 150 são idênticas às linhas 350 e 360.

O programa possui 103 linhas.

```
10 ' Chiptron
20 GOSUB 850
30 VZ=5:VX=VZ:PPX=0:PPZ=PX
40 COLOR15,15,15
50 SCREEN 2,,0
60 CIRCLE(128,96),90,1
70 PAINT(128,185),1,1
80 TIME=0
90 CIRCLE(128,36),10,15
100 CIRCLE(128,156),10,15
110 CIRCLE(168,96),20,15,,,.85
120 CIRCLE(88,96),20,15,,,.85
130 X=120:Y=96:Z=136:U=Y
140 PSET(X,Y),4
150 PSET(Z,U),6
160 DX=3:DZ=1
170 L$=INKEY$:IF L$="" THEN 260
180 IF L$="4" THEN DZ=1
190 IF L$="8" THEN DZ=2
200 IF L$="6" THEN DZ=3
210 IF L$="2" THEN DZ=4
220 IF L$="A" THEN DX=1
230 IF L$="W" THEN DX=2
240 IF L$="D" THEN DX=3
250 IF L$="X" THEN DX=4
260 IF DX=1 THEN X=X-1
270 IF DX=3 THEN X=X+1
280 IF DX=2 THEN Y=Y-1
290 IF DX=4 THEN Y=Y+1
300 IF DZ=1 THEN Z=Z-1
310 IF DZ=3 THEN Z=Z+1
320 IF DZ=2 THEN U=U-1
330 IF DZ=4 THEN U=U+1
340 PX=POINT(X,Y):PZ=POINT(Z,U)
350 PSET(X,Y),4
360 PSET(Z,U),6
370 IF PX<>1 THEN PUZ=PUZ+TIME+20:GOTO
400
380 IF PZ<>1 THEN PYX=PYX+TIME+20:GOTO
460
```

```
390 GOTO 170
400 FOR F=1 TO 20 STEP 3
410 CIRCLE(X,Y),F,RND(-TIME)*15
420 PLAY"S8T255L60GFA"
430 NEXT F
440 VX=VX-1:IF VX>0 THEN 520
450 GOTO 680
460 FOR F=1 TO 20 STEP 3
470 CIRCLE(Z,U),F,RND(-TIME)*15
480 PLAY"S8T255L60CAB"
490 NEXT F
500 VZ=VZ-1:IF VZ>0 THEN 520
510 GOTO 680
520 COLOR 15,13,13:KEY OFF
530 SCREEN 1
540 PRINT"          MSX- Chip Tron"
550 LOCATE 3,3
560 PRINT"Jogador 1: Vidas Pontos"
570 LOCATE 15,4
580 PRINT VX:LOCATE 20,4:PRINT PYX
590 LOCATE 3,9
600 PRINT"Jogador 2: Vidas Pontos"
610 LOCATE 15,10:PRINT VZ:LOCATE 20,10:
PRINT PUZ
620 LOCATE12,6:PRINT"TEMPO"
630 LOCATE12,7:PRINTTIME
640 LOCATE 1,22:PRINT"Aperte uma tecla
para cont."
650 IF INKEY$<>"" THEN 650
660 IF INKEY$="" THEN 660
670 GOTO 40
680 KEY OFF:FOR F=1 TO 14 STEP 2
690 COLOR F,F+1,F-1
700 NEXT F
710 COLOR 15,1,1
720 SCREEN1
730 PRINT"          MSX- Chip Tron"
740 LOCATE 5,10:PRINT"O jogador";
750 IF PYX>PUZ THEN PRINT" 1 venceu." E
LSE PRINT" 2 venceu."
```

```
760 PRINT:PRINT:PRINT"Jogador 1 fez"PYX
"pontos."
770 PRINT:PRINT"Jogador 2 fez"PUZ"ponto
s."
780 FOR F=1 TO 14
790 COLOR F,F-1,F+1:FOR G=0 TO 30:NEXT
G
800 NEXT F
810 PLAY"T60O3L24EBO4CO3ABGEBO4CO3ABGEC
O2A#O3CO3A#O3C"
820 IF INKEY$<>"" THEN 820
830 IF INKEY$="" THEN 830
840 RUN
850 COLOR 15,1,1
860 SCREEN 2
870 DRAW"S8BM65,20NR5U7R5BR3ND7D3R4NU3D
4BR3U7BR3ND7R3F2G2L3BU4"
880 DRAW"BR15R3ND7R3BR3ND7R3F2G2L2F3BU7
BR3D7R5U7NL5BR3ND7F7U7S6"
890 DRAW"BM130,150E7D7UH3GFBUBR6EU3HGD6
FRNE3R4NU6R2U3R3D3NL3BU2R"
900 DRAW"BR8BU3R4D8L4U4NR4U4BR7NR4D4R3F
D3GL3S4"
910 CIRCLE(50,145),30,7
920 PAINT(50,174),7,7
930 FOR F=0 TO 128
940 PSET(F,70),4:PSET(255-F,70),6
950 NEXT F
960 FOR F=1 TO 40 STEP 3
970 CIRCLE(128,70),F,RND(-TIME)*15
980 PLAY"S8T255L60EDGFA"
990 NEXT F
1000 IF INKEY$<>"" THEN GOTO 1000
1010 IF INKEY$="" THEN GOTO 1010
1020 PLAY"CEG"
1030 RETURN
```

ANÁLISE

O programa divide-se em três partes: apresentação, jogo e finalização.

A apresentação segue da linha 850 à 1030.

Nas linhas 850 e 860 definimos a cor e o tipo de tela a ser utilizado.

Nas linha 870 e 880 escrevemos CHIPTRON e na 890 e 900 a assinatura do autor e o ano (através da instrução DRAW).

Nas linhas 910 e 920 fazemos um círculo azul claro.

Da linha 930 à 950 movemos os dois TRONS e as linhas de 960 a 990 fazem a explosão do choque das motos.

As linhas 1000 e 1010 verificam se alguma tecla foi pressionada.

A linha 1020 emite um som e a 1030 retorna à rotina principal.

O jogo segue da linha 30 até 390.

Na linha 30 inicializamos as variáveis que irão contar os pontos e as vidas de ambos os jogadores.

Nas linhas 40 e 50 definimos a cor e o tipo de tela que usaremos.

Nas linhas 60 e 70 desenhamos a arena.

Na linha 80 inicializamos o tempo, zerando a variável do sistema TIME.

Da linha 90 à 120 desenhamos os obstáculos.

Na linha 130 inicializamos as variáveis responsáveis pelas coordenadas x e y de ambas as motos.

As linhas 140 e 150 colocam as motos na tela.

A linha 160 define a direção de cada moto.

A linha 170 verifica se alguma tecla foi pressionada. Em caso positivo ela desvia a execução para a rotina do teclado (linhas de 180 a 250), caso contrário a execução vai para a rotina de movimento (linhas de 260 a 330).

Na rotina do teclado verifica-se se alguma tecla direcional foi pressionada e em caso positivo altera-se a direção da moto.

Na rotina de movimento são verificadas as direções de ambas as motos e executados os movimentos.

Na linha 340 lemos o que há no espaço ocupado pelas motos.

Nas linhas 350 e 360 desenham-se as motos.

Na linha 370 e 380 verificamos se elas colidiram.

A linha 390 desvia a execução do programa para a inicialização.

Da linha 400 à 450 fazemos a explosão das motos, com um laço que determina o número de vezes e os raios das circunferências. A cor das mesmas é dada por RND (-TIME) * 15.

Na linha 420 é gerado o som e na 430 o final do laço.

Na linha 440 decrementamos as vidas e verificamos se chegou a zero. Caso seja zero o programa segue para

a rotina de finalização; caso contrário vai para o DISPLAY.

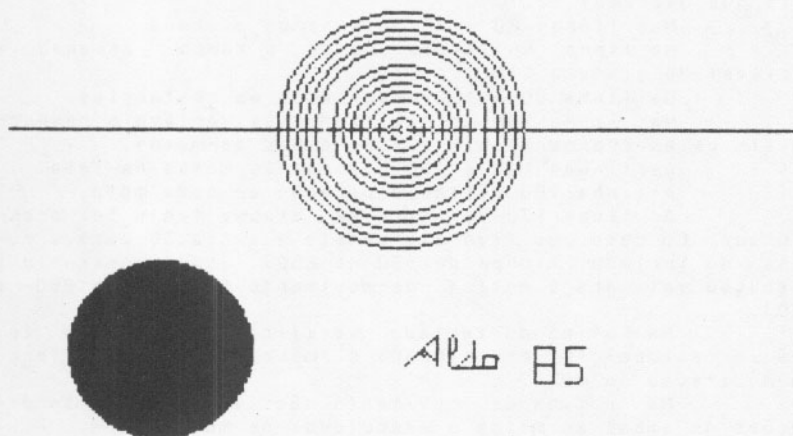
A forma desta rotina repete-se da linha 460 à 510.

Nas linha de 520 a 670 está a rotina do DISPLAY da situação do jogo, imprimindo na tela o número do jogador, seu número de vidas, seus pontos e o tempo decorrido.

A linha 670 devolve o programa para um novo confronto (montagem de tela e jogo).

Da linha 680 à 840 decorre a finalização.

Na linha 750 é decidido quem é o vencedor.

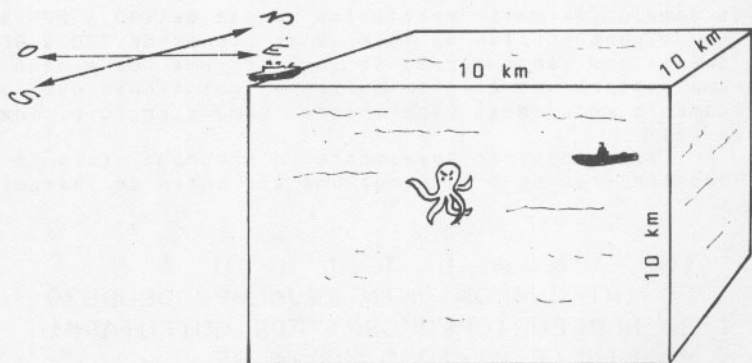
Na linha 850 reiniciamos desde a apresentação.

NAUTILUS

INSTRUÇÕES

Nas profundezas dos mares de um distante país vaga silenciosamente, espreitando e espionando, um submarino nuclear: o NÁUTILUS !

A única forma de acabar com a ameaça que ele representa é destruí-lo, e isso caberá a você, um importante almirante comandando pessoalmente o ataque a bordo de um destroyer.

O Náutilus está confinado pela topografia do relevo submarino a uma região de 10 km de largura, 10 km de comprimento e 10 km de profundidade (um cubo!). Seu destoyer encontra-se num dos vértices desse cubo (fig.10.1).

fig. 10.1

Você dispõe de dez cargas de profundidade e, para lançá-las, basta digitar as coordenadas das posições em que elas devem explodir, em relação à sua posição!

A coordenada X representa a distância ao longo da direção Norte-Sul.

A coordenada Y representa a distância ao longo da direção Leste-Oeste.

A coordenada Z representa a profundidade da explosão!

Cada vez que uma carga explode, devido à turbulência das águas, o Náutilus é forçado a ligar seus motores para estabilizar-se. Seus pequenos movimentos permitem que ele seja detectado por um sonar remoto lançado junto com a carga e que resiste à explosão.

Através dos dados enviados pelo sonar remoto, você pode saber a que distância (D) a carga explodiu do Náutilus. Sendo um almirante sábio, inteligente, astuto, lobo-do-mar mesmo, você não terá dificuldade em acertar o Náutilus depois de uns 6 ou 7 lançamentos.

Se por algum motivo um total de dez lançamentos forem feitos e o Náutilus não tiver sido destruído, você o será!!!

Nesse caso, um outro destroyer será designado para a tarefa!

Se o Náutilus for destruído, outro submarino será enviado para suas águas territoriais e nova batalha terá que ser travada.

Para reiniciar a batalha ou chamar outro destroyer, digite a barra de espaços.

Boa sorte, almirante!!!

DIGITAÇÃO

Este programa contém muitas linhas semelhantes e você pode economizar um bom tempo usando algumas para gerar as demais. As mais importantes são as de 140 a 270 que imprimem alguns sprites na tela, e as linhas de 770 a 860, que apenas armazenam dados. Se você der uma boa olhada no programa, notará que existem muitas outras linhas que são praticamente repetidas. Experimente, como exercício, identificá-las!

Após digitar totalmente o programa seja lá de que modo for, revise-o ao menos uma vêz antes de executá-lo.

```
10 ' N A U T I L U S
20 INTERVALON:ONINTERVAL=90GOSUB870
30 SCREEN2:OPEN"GRP:"FOR OUTPUTAS#1
40 LINE(0,0)-(255,70),4,BF
50 LINE(0,70)-(255,191),7,BF
```

```
60 FOR F=0 TO 15
70 C1=256*RND(1):C2=256*RND(1)
80 L=121*RND(-TIME)+70
90 LINE(C1,L)-(C2,L),3
100 NEXT F
110 FORG=1TO8:S$="":FORF=1TO8:READA$
120 S$=S$+CHR$(VAL("&H"+A$)):NEXTF
130 SPRITE$(G)=S$:NEXTG
140 PUT SPRITE 0,(40,63),1,1
150 PUT SPRITE 1,(48,63),1,2
160 PUT SPRITE 2,(98,170),4,3
170 PUT SPRITE 3,(106,170),4,4
180 PUT SPRITE 20,(210,25),11,5
190 PUT SPRITE 4,(190,120),8,6
200 PUT SPRITE 5,(150,10),15,7
210 PUT SPRITE 6,(158,10),15,8
220 PUT SPRITE 7,(110,12),15,7
230 PUT SPRITE 8,(118,12),15,8
240 PUT SPRITE 9,(180,20),15,7
250 PUT SPRITE 10,(188,20),15,8
260 PUT SPRITE 11,(200,5),15,7
270 PUT SPRITE 12,(208,5),15,8
280 G=98:FOR F=40 TO 210
290 PUT SPRITE 0,(F,63),1,1
300 PUT SPRITE 1,(F+8,63),1,2
310 IFF/2=F\2THEN G=G+1:PUTSPRITE 2,(G,
170),4,3:PUTSPRITE 3,(G+8,170),4,4
320 NEXT F
330 G1=150:G2=110:G3=180:G4=200
340 FOR F=100 TO -100 STEP -1
350 G1=G1-1:PUT SPRITE 5,(G1,10),15,7:P
UT SPRITE 6,(G1+7,10),15,8
360 G2=G2-1:PUT SPRITE 7,(G2,12),15,7:P
UT SPRITE 8,(G2+7,12),15,8
370 G3=G3-1:PUT SPRITE 9,(G3,20),15,7:P
UT SPRITE 10,(G3+7,20),15,8
380 G4=G4-1:PUT SPRITE 11,(G4,5),15,7:P
UT SPRITE 12,(G4+7,5),15,8
390 NEXT
400 COLOR1,5,5:SOUND8,10:SOUND9,9
```

```
410 LINE(0,42)-(154,191),5,BF
420 LINE(6,65)-(148,190),1,B
430 PRESET(30,52):PRINT#1,"NAUTILUS"
440 X0=INT(RND(-TIME)*10)
450 Y0=INT(RND(-TIME)*10)
460 Z0=INT(RND(-TIME)*10)
470 FOR T=1 TO 10:KK=70+T*10
480 PRESET(10,70):PRINT#1,"TIRO X Y Z
D":PRESET(14,KK):PRINT#1,T
490 X$=INKEY$:IFX$<"0"ORX$>"9"THEN490
500 X1=VAL(X$):PRESET(42,KK)
510 PRINT#1,X1
520 Y$=INKEY$:IFY$<"0"ORY$>"9"THEN520
530 Y1=VAL(Y$):PRESET(58,KK)
540 PRINT#1,Y1
550 Z$=INKEY$:IFZ$<"0"ORZ$>"9"THEN550
560 Z1=VAL(Z$):PRESET(74,KK)
570 PRINT#1,Z1
580 DX=X1-X0:DY=Y1-Y0:DZ=Z1-Z0
590 RESTORE860:INTERVALOFF:FORF=6TO13:R
EADX:SOUNDF,X:NEXTF
600 D=INT(.5+100*SQR(DX*DX+DY*DY+DZ*DZ)
)/100
610 PRESET(96,KK):PRINT#1,D
620 IF D=0 THEN 690
630 FORF=1TO30:NEXTF:INTERVALON:NEXTT
640 PRESET(20,2):PRINT#1,"Submarino se
aproximando!":LINE(20,30)-(130,40),4,BF
650 PRESET(10,30):PRINT#1,"Torpedo em n
ossa direcao!"
660 INTERVALOFF:FORF=1TO20:PLAY"O5L36F"
,"O5L36F":NEXTF:
670 FORG=0TO60:NEXTG:RESTORE850:FORF=6T
O13:READX:SOUNDF,X:NEXTF:FORF=1TO15:COL
ORF,F,F:PUTSPRITE0,(212,63),F,1:PUTSPRI
TE1,(220,63),F,2:NEXTF:COLOR1,5,5
680 IF STRIG(0)=0 THEN 670 ELSE CLOSE#1
:RESTORE 770:GOTO 10
690 IF 10-T>R THEN R=10-T
700 LINE(20,30)-(130,40),4,BF
```

```
710 PRESET(30,30):PRINT#1,"RECORDE:";R
720 LINE(26,44)-(120,62),1,BF:COLOR15,1
730 PRESET(40,54):PRINT#1,"PONTOS:";10-
T
740 FORG=0TO60:NEXTG:RESTORE850:FORF=6T
O13:READX:SOUNDF,X:NEXTF:FORF=1TO15:COL
ORF,F,F:PUTSPRITE2,(182,170),F,3:PUTSPR
ITE3,(190,170),F,4:NEXTF:COLOR1,5,5
750 PUTSPRITE2,(182,170),4,3:PUTSPRITE3
,(190,170),4,4
760 IFSTRIG(0)=0THEN740:ELSESOUND10,0:I
NTERVALON:GOTO400
770 DATA 00,00,00,03,0F,83,FF,3F
780 DATA 00,00,00,00,F0,FF,FE,FC
790 DATA 00,00,01,BF,7F,BF,00,00
800 DATA C0,80,E0,FE,FF,FC,00,00
810 DATA 3C,7E,FF,FF,FF,FF,7E,3C
820 DATA 18,3C,3C,18,36,59,A9,A5
830 DATA 1C,2B,DF,EF,FF,7D,1A,06
840 DATA 40,F8,34,FE,6F,FB,BC,78
850 DATA 20,220,0,0,15,100,60,0
860 DATA 20,220,15,15,15,100,60,0
870 PLAY"v10l12o5ao6l3g","v9l13o5ao6l3g
":RETURN
```

ANÁLISE

O funcionamento do NAUTILUS é bastante simples. Existe uma parte inicial do programa (linhas entre a 30 e a 390) que apenas "embelezam" a abertura visual, apresentando uma paisagem marítima e movendo pela tela alguns sprites (um polvo, núvens, o Sol, o Náutilus e o seu destroyer). Os dados usados para definição dos sprites estão armazenados nas linhas entre a 770 e a 840.

A linha 20 é a responsável pelo irritante som produzido pelo sonar do destroyer: sempre que o relógio interno do MSX avança um segundo e meio, a sub-rotina da linha 870 (um PLAY) é executada. Se você quiser eliminar esse som, elimine a linha 20. Se você quiser alterá-lo, altere as notas na instrução PLAY da linha 870.

A arte principal do jogo está entre as linhas 400 e 760. Vamos comentar rapidamente alguns trechos mais importantes.

As linhas entre a 440 e a 460 escolhem uma posi-

ção para o Nátilus, sorteando cada uma de suas coordenadas (XO,YO e ZO).

A linha 470 abre um laço que pode ser repetido dez vezes, uma para cada carga (tiro) que o destroyer pode lançar. As coordenadas de cada tiro são inseridas nas variáveis X1, Y1 e Z1 e impressas na tela pelas linhas entre a 490 e a 570.

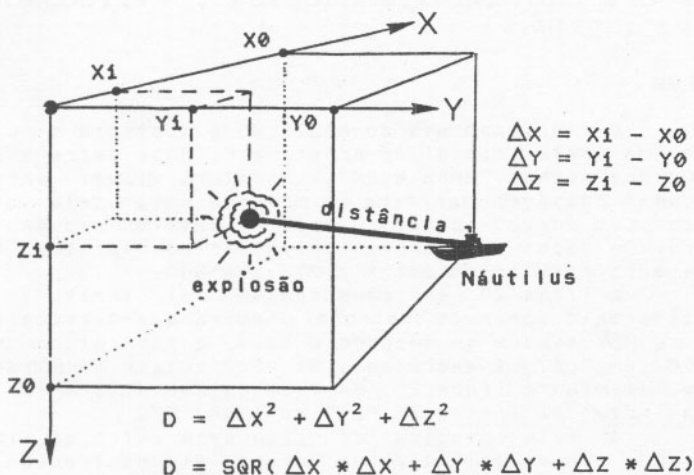
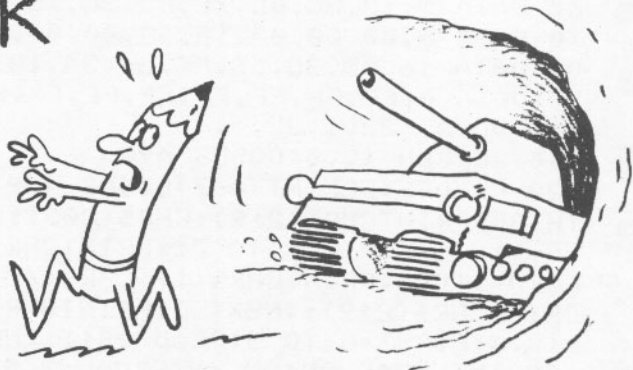
As linhas de 580 a 610 calculam a distância (D) da explosão até o submarino e geram um som de explosão (usando dados da linha 860).

Se o Náutilus foi atingido (D = 0), o programa é desviado para a linha 690, onde a rotina de finalização vitoriosa é executada (linhas de 690 a 750). Caso contrário, um novo lançamento (tiro) é realizado.

Ao terminarem os tiros, se o Nautilus não tiver sido atingido (D = 0), a rotina de finalização perdedora é executada (linhas de 630 à 670).

Em ambos os casos, digitando-se a barra de espaços (linhas 680 e 760) uma nova batalha é travada!

Propositadamente não entramos em detalhes sobre cada linha do programa, mas com um pouco de esforço você mesmo pode analisá-lo linha por linha. Suas rotinas são bastante simples e a mais complexa envolve apenas o teorema de "tritágoras" (fig. 10.2). Fica como exercício a alteração e complementação do programa por sua própria conta!

fig. 10.2

TANK

INSTRUÇÕES

Neste jogo sua missão é roubar um diamante que está guardado no fundo de um subterrâneo protegido por quatro robôs guardiães e em seguida trazê-lo de volta à superfície. Você dispõe de um tanque de guerra que pode abrir caminho a tiros e repelir os robôs que tentarão cercá-lo. Para pegar o diamante, basta passar com o tanque por cima dele.

O tanque é controlado pelas teclas de cursor e barra de espaço. As teclas ◁ e ▷ giram o tanque, a tecla △ o move para a frente e a barra de espaços é usada para atirar.

A saída fica no canto superior esquerdo da tela, sobre o lugar onde o tanque é colocado inicialmente. Não tente sair sem ter pego o diamante (você perceberá tardiamente que isto é desastroso e extremamente desaconselhável). Se mesmo assim você quiser experimentar, ao menos grave o programa antes !!!

DIGITAÇÃO

O programa não oferece dificuldade de digitação, mas é importante transcrever as linhas com bastante atenção para não haver surpresas na hora de rodar o programa.

```
1 '     TANK ATTACK
10 DATA 24,10,10,D6,D6,FE,FE,FE,C6
20 DATA 25,FC,FC,70,7F,70,FC,FC,00
30 DATA 26,C6,FE,FE,FE,D6,D6,10,10
40 DATA 27,3F,3F,0E,FE,0E,3F,3F,00
50 DATA 4,18,3C,3D,3E,3D,3C,18,3C
```

```
60 DATA 5,18,3C,BC,7C,BC,3C,18,3C
70 DATA 6,00,00,00,18,18,00,00,00
80 DATA 16,00,3C,56,FF,56,34,18,00
90 DATA 8,FF,FF,FF,FF,FF,FF,FF,FF
100 DATA -32,1,32,-1
110 SCREEN 1,,0:GOSUB 490
120 KEYOFF:DEFINT A-Z:GOSUB 450:CLS:WID
TH 30:PRINTCHR$(219);CHR$(203);STRING$(
28,219);:FOR I=1 TO 21:PRINTCHR$(219);:
FORJ=1TO28:PRINTCHR$(1);CHR$(72);:NEXT:
PRINTCHR$(219);:NEXT I:PRINTSTRING$(30,
219);:FOR I=0 TO 3:READ DS(I):NEXT
130 INTERVAL ON:ON INTERVAL=3 GOSUB 340
:DEF FNPO(X,Y)=6144+X+32*Y:VPOKE 6831,1
6
140 A=6178:D=2:F=0:P=0:VPOKE A,D+24:STR
IG(0)OFF:ON STRIG GOSUB 330:FOR I=1 TO
4:X(I)=15+I:Y(I)=15:NEXT I
150 STRIG(0)OFF:C=STICK(0):IF C=3THEND=
D+1ELSEIFC=7THEND=D-1ELSE170
160 OUT170,154:OUT170,26
170 IF D=-1THEND=3ELSEIFD=4THEND=0
180 VPOKE A,D+24:STRIG(0)ON
190 X=(A-6144)MOD32:Y=(A-6144)\32:FOR I
=1 TO 4:Z=8:IF I<4THENZ=32
200 VPOKEFNPO(X(I),Y(I)),Z:Z=4:DX=0:DY=
0:IFRND(1)>.5THENDX=SGN(X-X(I))ELSEDY=S
GN(Y-Y(I))
210 IF I<4THEN 250
220 IFVPEEK(FNPO(X(I)+DX,Y(I)))<>32THEN
IFVPEEK(FNPO(X(I)-DX,Y(I)))=32THENDX=-D
XELSEDX=0
230 IFVPEEK(FNPO(X(I),Y(I)+DY))<>32THEN
IFVPEEK(FNPO(X(I),Y(I)-DY))=32THENDY=-D
YELSEDY=0
240 U=FNPO(X(I),Y(I)):IF VPEEK(U+1)+VPE
EK(U-1)+VPEEK(U+32)+VPEEK(U-32)=32THENG
OSUB400
250 X(I)=X(I)+DX:Y(I)=Y(I)+DY:IFX(I)>XT
HENZ=5
```

```
260 IFVPEEK(FNPO(X(I),Y(I)))=6THENX(I)=
20:Y(I)=15
270 K=VPEEK(FNPO(X(I),Y(I))):IFK>23ANDK
<28THENT=0:STRIG(0)OFF:GOTO410ELSEVPOKE
FNPO(X(I),Y(I)),Z:NEXT I
280 K=VPEEK(DS(D)+A):IFC<>1THEN320
290 IFK<>8ANDK<>219THEN VPOKE A,32:A=A+
DS(D):VPOKE A,D+24:OUT 170,154:OUT 170,
26
300 IFK=16THENF=1:FORI=1TO10:BEEP:NEXTI
:ELSEIFK=4ORK=5THENT=0:STRIG(0)OFF:GOTO
410
310 IFK=203THENIFF=1THEN420ELSECLS:PRIN
T"VOCE NAO DEVIA TENTAR FUGIR...":NEW
320 GOTO 150
330 IF T>0 THEN RETURN ELSE T=A:DT=D:BE
EP:INTERVAL ON:RETURN
340 IF T=0 THEN RETURN
350 IF VPEEK(T)=6 THEN VPOKE T,32
360 T=T+DS(DT):IF VPEEK(T)=32 THEN VPOK
E T,6:RETURN
370 IF VPEEK(T)=8 THEN VPOKE T,32
380 IFVPEEK(T)=4ORVPEEK(T)=5THENGOSUB47
0
390 INTERVAL OFF:T=0:RETURN
400 U=FNPO(2,Y(I)):FOR J=U TO U+27:VPOK
E J,32:NEXT J:RETURN
410 CLS:LOCATE 0,10:PRINT"VOCE FOI CAPT
URADO":GOTO430
420 CLS:LOCATE 0,10:PRINT"VOCE CONSEGUI
U"
430 PRINT"JOGA DE NOVO ?";
440 A$=INPUT$(1):IFA$="S"THENRUNELSEIFA
$="N"THENENDELSE440
450 CLS:FOR I=0 TO 8:READ Z:FOR J=0 TO
7:READ B$:VPOKE J+8*Z,VAL("&H"+B$):NEXT
 J:VPOKE 6150+2*I,Z:NEXT I
460 I$=INPUT$(1):RETURN
470 FOR I=1TO4:IFFNPO(X(I),Y(I))<>TTHEN
NEXTI:RETURN
```

```
480 VPOKE T,32:X(I)=20:Y(I)=15:FORJ=1TO
5:OUT170,154:OUT170,26:NEXTJ:RETURN
490 VPOKE8192,135:VPOKE8193,96
500 VPOKE8194,246:VPOKE8195,39:FORI=819
6TO8210:VPOKEI,7:NEXTI:VPOKE8219,176
510 RETURN
```

ANÁLISE

O programa inicia chamando uma sub-rotina (linhas 450-510) que lê as instruções DATA do início do programa e faz a redefinição dos caracteres (tanque, robôs, diamante, etc).

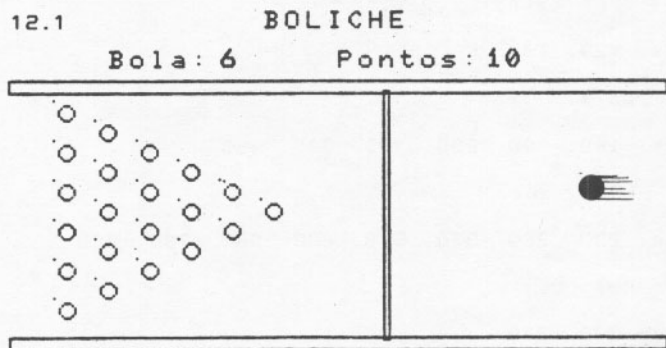
Em seguida, o programa imprime o cenário (linha 120) e inicia as variáveis (linhas 120-140). A ação central do jogo fica entre as linhas 330 e 390. A linha 400 serve para abrir uma passagem para o robô obturado (o que não escava túneis) e as linhas 410 à 440 encerram o jogo.

BOLICHE

INSTRUÇÕES

Este jogo é extremamente simples, a despeito de seu programa ser um tanto complexo.

Ao ser executado, ele gera na tela a imagem de uma pista de boliche vista de cima. Existem 21 garrafas que deverão ser derrubadas com 6 bolas.

fig. 12.1

A bola surge numa posição à direita da pista e vai se deslocando para a esquerda. Você pode movê-la para cima ou para baixo com as setas △ e ▽, porém apenas enquanto ela não chegou à faixa demarcatória central. Após esse limite, a bola segue uma linha reta.

Ao inserirmos este programa neste livro, não pensamos em suas qualidades como jogo, mas sim no exemplo que ele representa quanto à utilização da instrução ON SPRITE.

Como você deve ter notado, o programa parece um pouco longo para produzir algo tão simples. Esse aparente excesso de programa será compreendido mais adiante, na ANÁLISE.

DIGITAÇÃO

O programa é relativamente extenso e um bom tempo pode ser economizado através da re-utilização das linhas já introduzidas. Se você quiser, pode digitar o programa sequencialmente, começando da linha 10 e terminando na 800. Nós, entretanto, sugerimos que você utilize as linhas semelhantes. Para facilitar seu trabalho, relacionamos a seguir as linhas separadas em grupos.

* linhas: 40, 70, 100, 210, 220, 300, 360,

  FOR....=....TO....etc.

* linhas: 50, 80

  S$=S$+....

* linhas: 60, 90, 110

  SPRITE$(n)=....

* linhas: 120, 130

  DATA....

* linhas: 180, 190, 200, 320, 340, 450

  LINE....

* linhas: 230, 260, 370, 510, 600, 660, 730, 800

  PUT....

* linhas: 240, 270

  CIRCLE....

* linhas: 250, 410, 420, 430, 470, 500, 530, 570, 620, 680

  IF....THEN....

* linhas: 330, 350, 460

  PRESET....

* linhas: 540, 580, 590, 630, 640, 650, 690, 700, 710, 720, 750, 760, 770, 780, 790

IF....THEN....PUT SPRITE....

* linhas: 10, 30, 140, 290, 480, 520, 560, 610, 670, 740, Linhas REM

As linhas não inseridas em nenhum desses grupos são as seguintes: 10, 20, 150, 160, 170, 280, 310, 380, 390, 400, 440, 490, 550.

Após digitar completamente o programa, qualquer que seja a sequência escolhida, confira-o, ao menos uma vez, linha por linha.

```
10 ' B O L I C H E
20 SCREEN 2,0,0
30 '####################################
40 FOR F=1 TO 8:READ A$
50 S$=S$+CHR$(VAL("&H"+A$)):NEXT F
60 SPRITE$(1)=S$:S$=""
70 FOR F=1 TO 8:READ A$
80 S$=S$+CHR$(VAL("&H"+A$)):NEXT F
90 SPRITE$(2)=S$:S$=""
100 FOR F=1 TO 8:S$=S$+CHR$(0):NEXT F
110 SPRITE$(3)=S$
120 DATA 3C,66,FB,FD,FF,FF,7E,3C
130 DATA 38,44,BA,BA,BA,44,38,00
140 '####################################
150 MAXFILES=1:PT=0:COLOR 15,6,6:CLS
160 OPEN"GRP:" FOR OUTPUT AS #1
170 PRESET(60,2):PRINT #1,"      BOLICHE

          Bola:      Pontos:"
180 LINE(0,30)-(255,35),1,BF
190 LINE(0,134)-(255,139),1,BF
200 LINE(145,34)-(147,135),1,B:A=1
210 FOR K=0 TO 5:X=K*16+20
220 FOR Y=40+8*K TO 80 STEP 16
230 PUT SPRITE A,(X,Y),15,2:A=A+1
240 CIRCLE(X+2,Y+2),3,1:PAINT(X+2,Y+2),
1
250 IF Y=80 THEN 280
260 PUT SPRITE A,(X,160-Y),15,2:A=A+1
```

```
270 CIRCLE(X+2,162-Y),3,1:PAINT(X+2,162
-Y),1
280 NEXT Y,K
290 '##################################
300 FOR S=1 TO 6
310 Y=36+INT(RND(-TIME)*78)
320 LINE(78,19)-(90,28),6,BF
330 PRESET(70,19):PRINT#1,S
340 LINE(182,19)-(220,28),6,BF
350 PRESET(175,19):PRINT#1,PT
360 FOR X=255 TO 0 STEP-1
370 PUT SPRITE 0,(X,Y),1,1
380 SPRITE ON
390 ON SPRITE GOSUB 490
400 T=STICK(0)
410 IF X<145 THEN 440
420 IF T=5 THEN Y=Y+1:IF Y>125 THEN Y=1
25
430 IF T=1 THEN Y=Y-1:IF Y<35 THEN Y=35
440 NEXT X,S
450 LINE(182,19)-(220,28),6,BF
460 PRESET(175,19):PRINT#1,PT
470 IF STRIG(0)=-1 THEN RUN ELSE 470
480 '##################################
490 SPRITE OFF:BEEP:PT=PT+1
500 IF X<100 THEN 530
510 PUT SPRITE 21,(X,Y),0,3:RETURN
520 '##################################
530 IF X<84 THEN 570
540 IF Y<80 THEN PUT SPRITE 19,(X,Y),0,
3 ELSE PUT SPRITE 20,(X,Y),0,3
550 RETURN
560 '##################################
570 IF X<68 THEN 620
580 IF Y<72 THEN PUT SPRITE 16,(X,Y),0,
3:RETURN
590 IF Y<88 THEN PUT SPRITE 18,(X,Y),0,
3:RETURN
600 PUT SPRITE 17,(X,Y),0,3:RETURN
610 '##################################
```

```
620 IF X<52 THEN 680
630 IF Y<64 THEN PUT SPRITE 12,(X,Y),0,
3:RETURN
640 IF Y<80 THEN PUT SPRITE 14,(X,Y),0,
3:RETURN
650 IF Y<96 THEN PUT SPRITE 15,(X,Y),0,
3:RETURN
660 PUT SPRITE 13,(X,Y),0,3:RETURN
670 '################################
680 IF X<36 THEN 750
690 IF Y<56 THEN PUT SPRITE 7,(X,Y),0,3
:RETURN
700 IF Y<72 THEN PUT SPRITE 9,(X,Y),0,3
:RETURN
710 IF Y<88 THEN PUT SPRITE 11,(X,Y),0,
3:RETURN
720 IF Y<104 THEN PUT SPRITE 10,(X,Y),0
,3:RETURN
730 PUT SPRITE 8,(X,Y),0,3:RETURN
740 '################################
750 IF Y<48 THEN PUT SPRITE 1,(X,Y),0,3
:RETURN
760 IF Y<64 THEN PUT SPRITE 3,(X,Y),0,3
:RETURN
770 IF Y<80 THEN PUT SPRITE 5,(X,Y),0,3
:RETURN
780 IF Y<96 THEN PUT SPRITE 6,(X,Y),0,3
:RETURN
790 IF Y<112 THEN PUT SPRITE 4,(X,Y),0,
3:RETURN
800 PUT SPRITE 2,(X,Y),0,3:RETURN
```

ANÁLISE

Para facilitar a análise do programa, vamos dividi-lo em 4 blocos principais:

1) Bloco de definição de SPRITEs
   linha inicial- 40
   linha final- 130

2) Bloco da tela inicial
linha inicial- 150
linha final- 280

3) Bloco do jogo
linha inicial- 300
linha final- 470

4) Bloco de verificação das garrafas
linha inicial- 490
linha final- 800

Os três primeiros blocos são bastante simples e passaremos a descrevê-los sucintamente a seguir. Quanto ao quarto bloco, discutiremos um pouco mais detalhadamente seu funcionamento. É ele, na verdade, a parte complexa do programa, e se você conseguir entendê-lo, terá uma valiosa chave para criação de muitos outros programas!

O bloco de definição de SPRITEs gera três SPRITEs: um para as garrafas, outro para a bola e, o último, vazio.

Os códigos do SPRITE das garrafas estão armazenados em hexadecimal na linha 130, e os do SPRITE da bola na linha 120.

O SPRITE vazio é uma sequência de 8 códigos 0.

O bloco da tela inicial apresenta as mensagens, a pista e as garrafas no início do jogo. Note que para cada garrafa são impressos um SPRITE branco (linhas 230 e 260) e um círculo preto (linhas 240 e 270). A forma como isso é feito não é muito importante agora. Você mesmo, com um pouco de paciência, terá condições de entender.

Mais importante é notar que os SPRITEs das garrafas são impressos em função das variáveis X e Y. A sequência de posições geradas é dada na tabela a seguir e ilustrada na figura 12.2. Observe que as coordenadas de um SPRITE referem-se, na verdade, ao seu canto superior esquerdo!

fig. 12.2

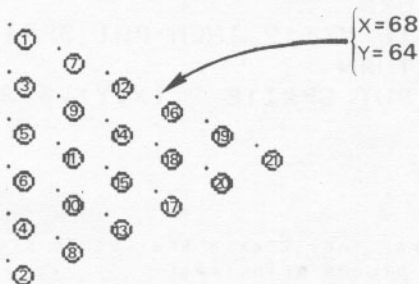



TABELA 12.1

X	Y	CAMADA
20	40	1
20	120	2
20	56	3
20	104	4
20	72	5
20	88	6
36	48	7
36	112	8
36	64	9
36	96	10
36	80	11
52	56	12
52	104	13
52	72	14
52	88	15
68	64	16
68	96	17
68	80	18
84	72	19
84	88	20
100	80	21

Vamos, agora, ao bloco do jogo.

Sua parte principal é um laço FOR...NEXT, repetido 6 vezes, uma para cada bola lançada. A linha 300 abre o laço. A linha 310 sorteia a posição vertical em que cada bola surgirá no lado direito da tela. As duas linhas seguintes, 320 e 330, apresentam, na tela, o número da bola que foi lançada. As linhas 340 e 350 apresentam os pontos do jogador.

A linha 360 inicia o laço que faz a bola ir se deslocando para a esquerda.

A linha 370 imprime a bola na tela. Observe que a sua posição depende das variáveis X e Y.

A linha 380 aciona os desvios produzidos pela linha 390.

Essa linha (390), desvia o programa para a subrotina da linha 490 sempre que a bola atingir alguma garrafa. Se não existisse a instrução SPRITE ON na linha 380, a 390 não funcionaria!

As quatro linhas seguintes permitem que a bola seja deslocada para cima e para baixo através das setas △ e ▽, enquanto ela estiver antes da linha limite.

A linha 440, fecha os laços abertos para as bolas e seus movimentos.

Finalmente, as linhas 450 e 460 apresentam a pontuação final do jogador e a linha 470 permite o reinício do jogo mediante o pressionamento da barra de espaços.

Chegamos, por fim, à sub-rotina que constitui o bloco de verificação das garrafas.

Quando a bola atinge uma garrafa, o programa deve apagá-la da tela. O problema está em determinar em qual camada está a garrafa atingida, pois isso é essencial para apagá-la.

Existem, inicialmente, 21 garrafas no vídeo, cada uma em uma camada (de 1 a 21).

A rotina a partir da linha 490 verifica a garrafa atingida através da posição da bola na tela.

Vejamos como isso é feito.

Quando a bola atinge alguma garrafa, ocorre uma sobreposição de SPRITEs e a linha 390 desvia o programa para a linha 490. Essa linha "desliga" o desvio produzido pela linha 390, gera um "beep" e incrementa a pontuação do jogador. Se a instrução SPRITE OFF fosse eliminada da linha 490, o desvio produzido pela linha 390 não seria desligado e o programa ficaria indefinidamente "parado" assim que alguma garrafa fosse atingida pela bola. Experimente eliminar a instrução SPRITE OFF para compreender melhor porque ela é essencial.

Logo depois, na linha 500, verifica-se se a garrafa mais à direita da tela foi atingida. Essa garrafa está na camada 21 e suas coordenadas são X=100 e Y=80. Se a coordenada X da bola for maior ou igual a 100, certamente a garrafa atingida foi a da camada 21 (veja a fig. 12.2).

Caso a coordenada X da bola seja menor que 100, alguma outra garrafa foi atingida e o programa vai para a linha 530.

Se a coordenada X da bola for maior ou igual a 84, significa que a garrafa atingida foi a da camada 19 ou da camada 20. Nesse caso, a decisão é feita com base na coordenada Y da bola. Se Y é menor que 80, a garrafa atingida é a da camada 19, e se Y for maior ou igual a 80, a garrafa é a da camada 20.

Se a coordenada X da bola for menor que 84, o programa é desviado para a linha 570, pois a garrafa atingida foi alguma outra.

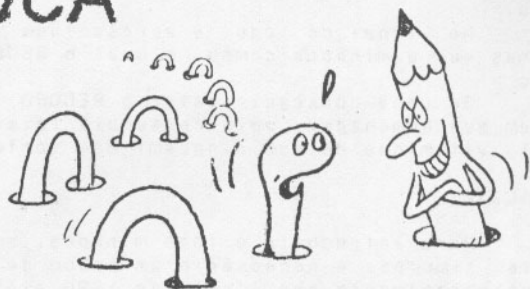
Assim, o programa prossegue na sub-rotina, averiguando garrafa por garrafa, até achar a que foi atingida. Então retorna à rotina principal.

Essa versão do BOLICHE necessita de todas essas linhas porque usa SPRITEs. Um efeito mais modesto é obtido com o programa a seguir que utiliza a tela de baixa resolução.

```
10 'B O L I C H E  II
20 COLOR 15,6,6:SCREEN3
30 OPEN"GRP:"FOROUTPUTAS#1
40 PRESET(24,0):PRINT#1,"BOLICHE":PT=0
50 LINE(0,30)-(255,35),1,BF
```

```
60 LINE(0,128)-(255,133),1,BF
70 FOR K=0 TO 5:X=K*16+20
80 FOR Y=40+8*KTO80STEP16:PSET(X,Y)
90 IF Y<>80 THEN PSET(X,160-Y)
100 NEXT Y,K
110 FOR S=1 TO 11:A%=32000!*RND(-TIME)
120 LINE(10,150)-(90,191),6,BF
130 PRESET(10,150):PRINT#1,S
140 LINE(140,150)-(250,191),6,BF
150 PRESET(140,150):PRINT#1,PT
160 Y=36+4*INT(RND(A%)*20)
170 FOR X=255 TO 0 STEP-4
180 IF POINT(X,Y)=15 THEN PT=PT+1
190 FORF=1TO5:NEXTF
200 PSET(X,Y),1:PRESET(X+4,J)
210 T=STICK(0):J=Y
220 IFT=5THENY=Y+4:IFY>124THENY=124
230 IFT=1THENY=Y-4:IFY<36THENY=36
240 NEXT X,S
250 LINE(140,150)-(250,191),6,BF
260 PRESET(140,150):PRINT#1,PT
270 IF STRIG(0)=0 THEN 270 ELSE RUN
```

Muitas alterações podem ser feitas no programa que usa SPRITEs a fim de melhorá-lo. Uma delas pode ser a substituição do "beep" da linha 490 pelo barulho de choque característico do boliche. Outra alteração seria a opção de mais de um jogador, com a indicação do recorde e dos nomes. Espaço na tela há de sobra para isso!

Com um pouco de imaginação, você mesmo pode implementar seu programa!

MINHOCA

INSTRUÇÕES

Neste jogo, você guia uma minhoca muito faminta que precisa comer vitaminas para poder continuar viva.

A cada vitamina que é devorada, a minhoca aumenta suas energias e, em consequência, cresce um pouco.

Se a minhoca não comer as vitaminas, ela acaba morrendo de desnutrição. E é por isso que a minhoca nunca para de andar, sempre à procura das preciosas vitaminas que estão espalhadas pelo caminho.

Você deve guiá-la de maneira a fazer com que ela vá ao encontro de qualquer uma das quatro vitaminas que foram colocadas aleatoriamente na tela.

Se a minhoca morder a si mesma ou a uma das quatro paredes laterais, morre e o jogo termina. Cuidado ao guiá-la! Ela só pode comer vitaminas e deve fazer isso o mais rápido possível. A cada passo dado, um pouco de energia é consumida e a minhoca pode morrer de fome!

Para guiar a minhoca use as setas que estão na parte inferior à direita no teclado. Acionando simultaneamente duas teclas você pode guiá-la também nas direções diagonais.

Na parte inferior da tela você verá uma linha que indica quanto a minhoca tem de energia. Inicialmente ela vai do canto esquerdo ao direito. Com a perda da energia ela diminui da direita para a esquerda.

Simultaneamente ao movimento da minhoca, um alarme sonoro é regularmente emitido, indicando através da frequencia a quantidade de energia. Quanto mais agudo, menos energia a minhoca possui.

Quanto mais vitaminas forem abocanhadas, mais

-.

tempo a minhoca permanecerá viva e mais tempo você fica jogando.

No final do jogo, é apresentada a quantidade de vitaminas que a minhoca comeu e qual o RECORD até aquele momento.

Se você conseguir bater o RECORD, ouvirá uma música em sua homenagem, pois conseguiu fazer a minhoca comer mais vitaminas do que ninguém! Boa sorte!

DIGITAÇÃO

Para introduzir o jogo minhoca, embora ele seja bastante simples, é necessário um pouco de atenção e cuidado, principalmente nas linhas de 1980 até 2150, onde os caracteres estão juntos e a leitura é um pouco dificultada.

Para começar a digitar, comande AUTO 1000 (e tecle RETURN). Assim o Expert vai numerando as linhas automaticamente, simplificando muito o trabalho.

Uma tática inteligente para uma boa digitação é ir conferindo cada linha digitada antes de teclar RETURN. Assim, ao terminar uma linha, confira-a e corrija os eventuais erros detectados. Com isso, evitam-se desagradáveis surpresas ao executar o programa.

Após terminar a digitação, convém gravar o programa em fita se você pretende usá-lo posteriormente. Digite o comando CSAVE"minhoc", prepare o gravador e comande RETURN .

Quando aparecer o "Ok" na tela, rode o programa e comece a jogar.

```
1000 '----------------------------
1010 '    JOGO     MINHOCA
1020 '
1030 '   RUBENS    DEZ / 85
1040 '----------------------------
1050 CLEAR 3000
1060 DEFINT A-Z
1070 MA = 10
1080 SCREEN 1,0,0
1090 WIDTH 32
1100 COLOR 15,1,1
1110 CLS
1120 KEY OFF
1130 LOCATE 7,1,0
1140 PRINT ">>>  MINHOCA  <<<"
1150 LOCATE 0,4
```

```
1160 PRINT "Voce deve guiar uma minhoca
 de"
1170 PRINT "maneira a faze-la comer o m
aior"
1180 PRINT "numero possivel de vitamina
s."
1190 LOCATE 0,9
1200 PRINT "A cada vitamina que e comid
a, a"
1210 PRINT "minhoca cresce um pouco."
1220 LOCATE 0,13
1230 PRINT "Para guiar a minhoca,utiliz
e as"
1240 PRINT "setas do lado direito do te
clado"
1250 LOCATE 5,22
1260 PRINT "TECLE QUALQUER TECLA ";
1270 IF INKEY$ <> "" THEN GOTO 1270
1280 X$ = INPUT$(1)
1290 '
1300 '  rotina principal
1310 '
1320 GOSÚB 1580
1330 INTERVAL OFF
1340 SOUND 7,255
1350 FOR I = 1 TO 500
1360 NEXT I
1370 SCREEN 1
1380 LOCATE 5,5
1390 PRINT "Voce comeu ";
1400 PRINT VT;" vitaminas"
1410 LOCATE 5,8
1420 PRINT "o record e ";
1430 PRINT MA
1440 IF VT > MA THEN MA = VT
1450 IF MA > VT THEN GOTO 1250
1460 LOCATE 0,13
1470 PRINT "PARABENS !! BATEU O RECORD
!!!"
1480 X$="T150L4"
```

```
1490 SOUND 7,248
1500 PLAY X$,X$,X$
1510 PLAY "O4B","O5D",""
1520 PLAY "O5C","O5E",""
1530 PLAY "O4B","O5D","O5G"
1540 PLAY "O4A","O5D","O5F#"
1550 PLAY "O3G","O4B","O5G"
1560 IF PLAY(1) THEN GOTO 1560
1570 GOTO 1250
1580 '
1590 '  rotina do jogo
1600 '
1610 SCREEN 3
1620 BD$= "r250d176l250u176"
1630 PSET (0,0),3
1640 DRAW BD$
1650 FOR X = 12 TO 243 STEP 4
1660     PSET (X,187),11
1670     SOUND 0,X-12
1680     SOUND 1,0
1690     SOUND 2,X
1700     SOUND 3,0
1710     SOUND 4,X+12
1720     SOUND 5,0
1730     SOUND 7,248
1740     SOUND 8,5
1750     SOUND 9,5
1760     SOUND 10,5
1770 NEXT X
1780 CT = 0
1790 XR = 20
1800 YR = 100
1810 XC = 24
1820 YC = 100
1830 XT = 243
1840 ON INTERVAL=10 GOSUB 2250
1850 INTERVAL ON
1860 PSET (XR,YR),4
1870 PSET (XC,YC),7
1880 ID$ = CHR$(3)
```

```
1890 DR = 3
1900 FOR I = 1 TO 4
1910      GOSUB 2160
1920 NEXT I
1930 VT=0
1940 PSET (XC,YC),4
1950 IF XT < 12 THEN RETURN
1960 X$ = INKEY$
1970 X = STICK(0)
1980 IF X <> 0 THEN DR = X
1990 IF DR>1 AND DR<5 THEN XC=XC+4
2000 IF DR>5 AND DR<9 THEN XC=XC-4
2010 IF DR=8 OR DR=1 OR DR=2 THEN YC=YC
-4
2020 IF DR>3 AND DR< 7 THEN YC=YC+4
2030 IF POINT(XC,YC)=2 THEN GOSUB 2400
2040 IF POINT(XC,YC) <> 1 THEN RETURN
2050 PSET (XC,YC),7
2060 ID$ = ID$ + CHR$(DR)
2070 IF CT > 0 THEN CT=CT-1 : GOTO 1940
2080 PRESET (XR,YR)
2090 X = ASC(LEFT$(ID$,1))
2100 IF X>1 AND X<5 THEN XR=XR+4
2110 IF X>5 AND X<9 THEN XR=XR-4
2120 IF X=8 OR X=1 OR X=2 THEN YR=YR-4
2130 IF X>3 AND X<7 THEN YR=YR+4
2140 ID$ = MID$(ID$,2)
2150 GOTO 1940
2160 '
2170 '  poe comida no jogo
2180 '
2190 X = INT(220*RND(1))+12
2200 Y = INT(160*RND(1))+12
2210 IF POINT(X,Y) <> 1 THEN GOTO 2190
2220 PSET (X,Y),2
2230 VT=VT+1
2240 RETURN
2250 '
2260 '  rotina de tempo
2270 '
```

```
2280 INTERVAL STOP
2290 SOUND 0,255
2300 SOUND 1,XT/16
2310 SOUND 7,254
2320 SOUND 8,16
2330 SOUND 11,0
2340 SOUND 12,5
2350 SOUND 13,0
2360 XT = XT - 1
2370 PSET (XT,187),1
2380 INTERVAL ON
2390 RETURN
2400 '
2410 '  COMEU UMA VITAMINA
2420 '
2430 INTERVAL STOP
2440 SOUND 2,0
2450 FOR X = 15 TO 0 STEP -1
2460      SOUND 3,X
2470      SOUND 7,253
2480      SOUND 9,10
2490 NEXT X
2500 SOUND 7,255
2510 GOSUB 2160
2520 PRESET (XC,YC)
2530 IF LEN(ID$)<240 THEN CT=CT+10 ELSE
 CT=0
2540 FOR X=1 TO 4
2550      IF XT>240 THEN GOTO 2580
2560      PSET (XT,187),11
2570      XT=XT+4
2580 NEXT X
2590 INTERVAL ON
2600 RETURN
```

ANÁLISE

Se você quiser fazer algumas modificações no programa, aqui vão algumas dicas.

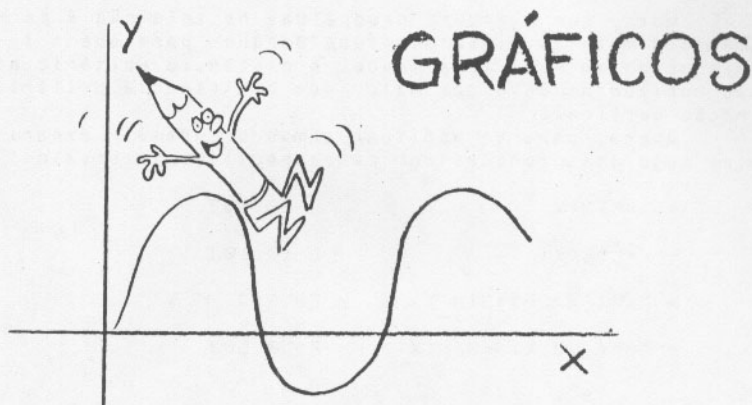
Na linha 1070 está definido o valor do RECORD quando o jogo começa. Se você quiser, pode alterar o valor 10 para outro menor, tornando mais fácil a quebra do RECORD.

As instruções do jogo estão entre as linhas 1120 e 1260. Altere-as como quiser, ou elimine-as para abreviar o tempo de gravação.

Desde a linha 1460 até a linha 1560 está a rotina que o parabeniza por bater o RECORD. Você pode colocar mais efeitos ou então modificar a música.

O número de vitaminas que aparecem no início de uma partida está definido na linha 1900. Quanto mais vitaminas você colocar, mais fácil se torna o jogo.

Na linha 1840 está o ritmo no qual a minhoca perde energia. Se você o alterar para 60, por exemplo, a minhoca demora mais tempo para ficar sem energia.

Usando a imaginação, muitas outras alterações poderão ser feitas.

INSTRUÇÕES

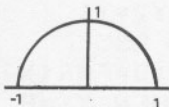
A tela de alta resolução do MSX não serve apenas para se fazer jogos com desenhos detalhados. O programa apresentado a seguir a utiliza para traçar o gráfico de qualquer função de uma variável previamente definida pelo usuário, dentro de um certo intervalo.

A função deve ser definida na linha 30 do programa e a variável independente deve ser necessariamente simbolizada por X.

A título de exemplo usamos a linha

```
30 DEFFNR(X)=SQR(1-X*X)
```

que corresponde à equação de uma semi-circunferência de raio unitário.

fig. 14.1

Quando executado, o programa pede a introdução do menor e do maior valor que a variável independente, X, deverá assumir. Ao introduzir esses valores, deve-se tomar muito cuidado para não tornar o cálculo da função inviável, gerando erros do tipo

```
Illegal function call, Overflow,
Division by zero, Type mismatch.
```

Em nosso exemplo, o intervalo válido para X está entre -1 e 1. Experimente introduzir dois valores quaisquer dentro desses limites. Por exemplo, 0 e 1. O programa deverá construir o gráfico na tela.

Note que a figura produzida na tela não é perfeitamente simétrica. Isso acontece porque, para que a tela seja usada em sua totalidade, a distância unitária na direção horizontal deve ser maior que a distância unitária na direção vertical.

Agora, para se habituar com o programa, experimente as seguintes funções com os respectivos intervalos.

* X*SIN(X) [-10,10]

* X*COS(X) [-10,10]

* EXP(-X/20)*SIN(X) [0,90]

* SQR(ABS(X))*SIN(X) [-50,50]

DIGITAÇÃO

Não há nenhuma observação importante a ser feita com relação à digitação das linhas deste programa.

Se você quiser economizar tempo, lembre-se de utilizar as linhas já introduzidas para gerar outras linhas semelhantes.

```
10 ' Graficos em alta resolucao
20 SCREEN 0
30 DEF FNR(X)=SQR(1-X*X)
40 INPUT"Menor X";X1
50 INPUT"Maior X";X2
60 SCREEN 2
70 X=X1:D1=(X2-X1)/255
80 Y1=FNR(X):Y2=Y1
90 FOR F=0 TO 255
100 X=X1+F*D1:B=FNR(X)
110 IF B<Y1 THEN Y1=B
120 IF B>Y2 THEN Y2=B
130 NEXT F
140 D2=(Y2-Y1)/191
150 IF D2=0 THEN D2=1
160 K=-X1/D1:L=-Y1/D2
170 IF Y1>0 THEN L=0
180 IF Y2<0 THEN L=191
190 IF X1>0 THEN K=0
200 IF X2<0 THEN K=255
210 FOR F=0 TO 255
220 X=X1+F*D1
```

```
230 R=191-INT(.5+(FNR(X)-Y1)/D2)
240 IF F=0 THEN S=F:T=R
250 LINE(S,T)-(F,R):PSET(F,191-L)
260 S=F:T=R
270 IF F>=0 AND F<=191 THEN PSET(K,191-
F)
280 NEXT F
290 GOTO 290
```

ANÁLISE

A compreensão do funcionamento do programa é muito mais fácil para aqueles que têm uma certa formação matemática. Se esse não for o seu caso, não se surpreenda por não entendê-lo!

Na verdade o que ele faz é bastante simples! Apenas constrói o gráfico de uma função usando inteiramente a tela.

A linha 30, como já sabemos, é onde a função deve ser definida. As linhas 40 e 50 delimitam o intervalo em que o gráfico deve ser construido.

A linha 70 define o valor D1 que será usado como passo (ou intervalo) das abscissas. Esse valor é calculado de modo que toda a extensão horizontal da tela seja usada. Depois, da linha 80 à linha 130, calcula-se o intervalo da imagem da função, isto é, o menor e o maior valor que ela vai assumir quando a variável independente X varrer o intervalo determinado nas linhas 40 e 50.

As linhas 140 e 150 definem o fator de escala D2 de modo a fazer com que toda a extensão vertical da tela seja usada.

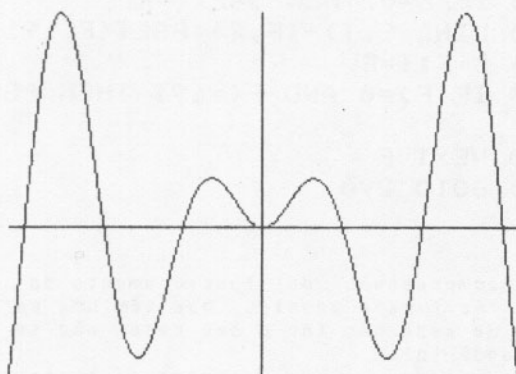
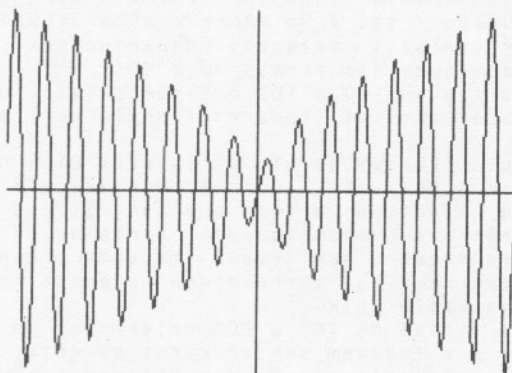
Note que, em geral, os fatores D1 e D2 são diferentes, pois a tela não é um quadrado mas sim um retângulo. Em parte, é devido a isso que os gráficos produzidos não são simétricos, mesmo quando seria de se esperar que fossem! Outro fator a ser levado em conta é que o intervalo de variação de Y não corresponde necessariamente ao intervalo de variação de X.

As linhas de 160 à 200 determinam as posições em que os eixos X e Y devem ser traçados na tela. Essas posições dependem respectivamente dos valores L e K.

As linhas de 210 à 280 calculam o valor da função e o ponto correspondente da tela para cada valor de X entre 0 e 255. Esses pontos são ligados por linhas, tornando o traçado do gráfico contínuo.

Se você quiser um gráfico descontínuo, experimente alterar a linha 250 para:

```
250 PSET(F,R):PSET(F,191-L)
```

FUNÇÃO SQR(ABS(X)*SIN(X))

INSTRUÇÕES

Uma das mais interessantes aplicações dos computadores está na produção e manipulação de figuras tri-dimensionais. Através disso, podemos simular formas e movimentos, analisando-os numa tela de vídeo.

Os máximos expoentes que essas técnicas produziram são conhecidos de todos nós: as "vinhetas eletrônicas", como as da televisão (blim,blim), por exemplo.

Essas belas imagens que são apresentadas nos intervalos comerciais das TV's são produzidas por grandes computadores. A memória usada é imensa e a técnica envolve linguágens refinadas, especificamente desenvolvidas para confecção e movimentação de figuras. Além disso, a velocidade de processamento é extremamente elevada!

Nos micros, sem distinção, existem limitações inerentes à arquitetura da eletrônica empregada. Um MICROCOMPUTADOR é fundamentalmente diferente de um COMPUTADOR. A capacidade de memória e a velocidade são muito menores e a linguagem empregada é, em geral, o BASIC. Em alguns casos, um único comando de uma linguagem específica para traçados gráficos corresponde a dezenas de linhas do BASIC!

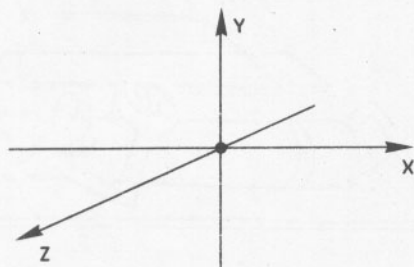
Desconsiderar as diferenças seria o mesmo que não fazer distinção entre uma caneta e uma máquina de escrever: ambas servem para escrever em papel, ambas são operadas através dos dedos, etc...

Apesar de tudo, o BASIC MSX nos permite algumas modestas experiências nessa área que, no entanto, já representam um primeiro passo. O programa que apresentamos é capaz de gerar e movimentar, no vídeo, uma figura qualquer, definida previamente pelo usuário.

Para definir a figura, basta introduzir seus

vértices e indicar como eles devem ser ligados.

Obviamente, é essencial a escolha de um sistema de coordenadas adequado. Vamos, desde já, convencionar o uso do sistema cartesiano tri-ortogonal representado na figura 15.1.

fig. 15.1

No programa, o plano XY é o plano da tela. O eixo Z é, portanto, perpendicular à tela e orientado como se estivesse saindo dela em direção ao usuário.

Um CUBO, nesse sistema, pode ser representado pelo conjunto de vértices:

V1=(1, 1, 1)	V2=(1, 1,-1)
V3=(1,-1, 1)	V4=(1,-1,-1)
V5=(-1, 1, 1)	V6=(-1, 1,-1)
V7=(-1,-1, 1)	V8=(-1,-1,-1)

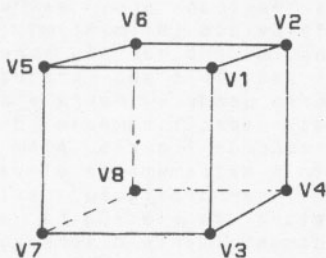
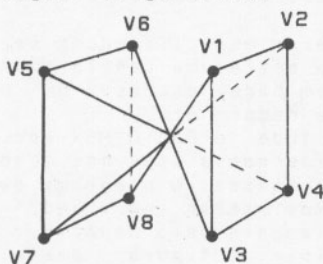
fig. 15.2

Entretanto, os mesmos vértices podem também representar outra figura (fig. 15.3).

fig. 15.3

Existe o seguinte problema: dado um conjunto de pontos, como interligá-los? Veja a figura 15.4.

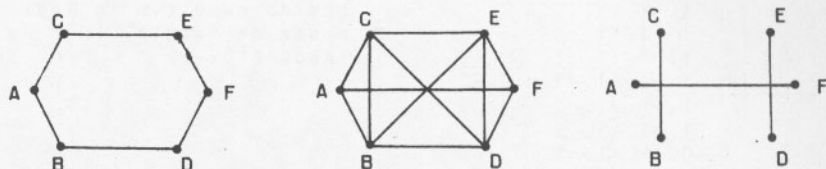



fig. 15.4

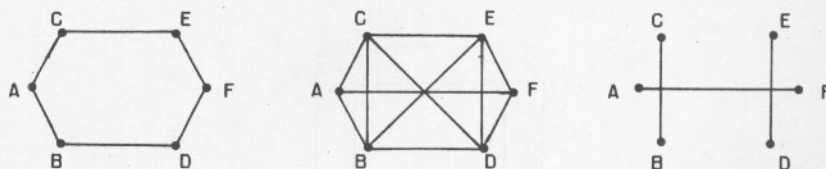
Note que os pontos são os mesmos e existem vários modos de ligá-los! Necessitamos, pois, de uma notação que nos indique isso!

Vamos usar a seguinte convenção:

* Colocaremos um algarismo 0 à esquerda do ponto se alguma linha partir dele (PONTO).

* Colocaremos um algarismo 1 à esquerda do ponto se alguma linha chegar até ele (TRAÇO).

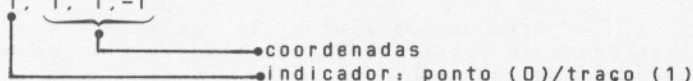
Veja a figura 15.5. Cada figura é descrita, através dessa convenção, de uma forma distinta. Apesar de seus vértices serem os mesmos, evita-se qualquer confusão!

fig. 15.5

0A1B1D1F1E1C1A / 0A1B1C1A1F1E1C1D1E1B1D1F / 0A1F0B1C0D1E

Agora não há mais como se atrapalhar! Essa notação é eficaz e será usada no programa!

Vamos, novamente, representar um CUBO. Cada um de seus pontos será representado por 4 dígitos. O primeiro é o indicador ponto/traço (0 para ponto e 1 para traço!). Os outros três são as coordenadas do ponto naquele nosso sistema de coordenadas cartesianas.

```
0, 1, 1, 1
1, 1, 1,-1
1, 1,-1,-1
1, 1,-1, 1
1, 1, 1, 1
1,-1, 1, 1
1,-1, 1,-1
1,-1,-1,-1
1,-1,-1, 1
1,-1, 1, 1
0,-1,-1,-1
1, 1,-1,-1
0,-1,-1, 1
1, 1,-1, 1
0,-1, 1,-1
1, 1, 1,-1
│  └──┬──┘
│     └────────●coordenadas
└──────────────●indicador: ponto (0)/traço (1)
```

Representação dos vértices do cubo com o indicador de ponto (0) ou traço (1).

É exatamente desse jeito que se devem inserir os pontos no programa. Imagine que você queira introduzir os pontos listados anteriormente. Antes de mais nada, é necessário atribuir à variável N% (na linha 30) o número de pontos que serão introduzidos.

Em nosso exemplo (na listagem), N%=12.

Para introduzir o CUBO deve-se fazer N%=16. Depois basta introduzir os pontos a partir da linha 570.

A sequência dos pontos é fundamental e deve-se tomar muito cuidado para não alterá-la!

Com os pontos já colocados nas instruções DATA's, basta comandar RUN (ou F5) e o cubo surge na tela!

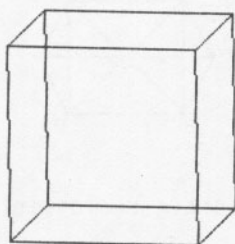
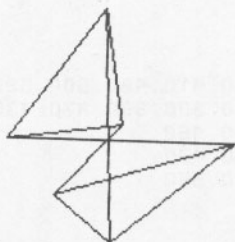
fig. 15.6

Antes de tentar colocar o CUBO no programa, experimente executá-lo como ele está na listagem. Devem surgir na tela duas pirâmides ligadas por um vértice comum.

fig. 15.7

Existem 8 teclas (de F1 à F8) que permitem mover ou alterar a figura na tela. A seguir, descrevemos a função de cada uma delas:

- F1 – Gira a figura no sentido direto ao redor do eixo Y.
- F2 – Gira a figura no sentido retrógrado ao redor do eixo Y.
- F3 – Gira a figura no sentido direto ao redor do eixo X.
- F4 – Gira a figura no sentido retrógrado ao redor do eixo X.
- F5 – Afasta a figura, diminuindo suas coordenadas ao longo do eixo Z.
- F6 – Aproxima a figura, aumentando suas coordenadas ao longo do eixo Z.
- F7 – Aumenta o efeito de perspectiva.
- F8 – Diminui o efeito de perspectiva.

Experimente algumas vezes cada uma dessas teclas. Depois, tente criar e introduzir outras figuras (o CUBO, por exemplo!).

DIGITAÇÃO

O programa não oferece dificuldades de digitação. Ele é curto e, em geral, há apenas uma instrução em cada linha.

Se você quiser economizar tempo, utilize as linhas já introduzidas para produzir outras. A seguir, dividimos as linhas semelhantes em grupos.

```
*10
*20
*30
*40,80,50
*60
*70,100,140,210,280,350,420
*90
*110,120
*130,170,250,320,390,460
```

```
*150,160
*180
*190
*200,270,340,410,480,500,520,540,560
*220,230,290,300,360,370,430,440
*240,310,380,450
*260,330,400,470
*490,510,530,550
*570, ...
```

```
10 '    Perspectivas Tri-Dimensionais
20 FOR F=1 TO 8:KEY(F)ON:NEXT F
30 N%=12:COLOR1,9,8:SCREEN2
40 DIM I(N%),X(N%),Y(N%),Z(N%)
50 DIM S(N%),T(N%):J%=5:K%=1:E%=100
60 S=SIN(.2):C=COS(.2)
70 FOR F%=1 TO N%
80 READ I(F%),X(F%),Y(F%),Z(F%)
90 Z(F%)=-Z(F%):NEXT F%
100 FOR F%=1 TO N%
110 S(F%)=120+X(F%)*E%/(K%*Z(F%)+J%)
120 T(F%)=86-Y(F%)*E%/(K%*Z(F%)+J%)
130 NEXT F%:CLS
140 FOR F%=1 TO N%
150 IF I(F%)=0 THEN PSET(S(F%),T(F%))
160 IF I(F%)=1 THEN LINE(S(F%),T(F%))-(
S(F%-1),T(F%-1))
170 NEXT F%
180 ON KEY GOSUB 200,270,340,410,480,50
0,520,540
190 GOTO 180
200 ' Gira a frente para a direita
210 FOR F%=1 TO N%
220 X=X(F%)*C-Z(F%)*S
230 Z=Z(F%)*C+X(F%)*S
240 X(F%)=X:Z(F%)=Z
250 NEXT F%
260 RETURN 100
270 ' Gira a frente para a esquerda
280 FOR F%=1 TO N%
290 X=X(F%)*C+Z(F%)*S
300 Z=Z(F%)*C-X(F%)*S
```

```
310 X(F%)=X:Z(F%)=Z
320 NEXT F%
330 RETURN 100
340 ' Gira a frente para baixo
350 FOR F%=1 TO N%
360 Y=Y(F%)*C+Z(F%)*S
370 Z=Z(F%)*C-Y(F%)*S
380 Y(F%)=Y:Z(F%)=Z
390 NEXT F%
400 RETURN 100
410 ' Gira a frente para cima
420 FOR F%=1 TO N%
430 Y=Y(F%)*C-Z(F%)*S
440 Z=Z(F%)*C+Y(F%)*S
450 Y(F%)=Y:Z(F%)=Z
460 NEXT F%
470 RETURN 100
480 ' Afasta
490 J%=J%*2:RETURN 100
500 ' Aproxima
510 J%=J%/2:RETURN 100
520 ' Aumenta a perspectiva
530 K%=K%*2:RETURN 100
540 ' Diminui a perspectiva
550 K%=K%/2:RETURN 100
560 ' Dados =)(p/t,x,y,z)
570 DATA 0 ,-3 , 0 , 0
580 DATA 1 , 0 , 0 ,-3
590 DATA 1 , 0 ,-3 , 0
600 DATA 1 ,-3 , 0 , 0
610 DATA 1 , 3 , 0 , 0
620 DATA 1 , 0 , 0 , 3
630 DATA 1 , 0 , 3 , 0
640 DATA 1 , 3 , 0 , 0
650 DATA 0 , 0 , 0 , 3
660 DATA 1 , 0 , 0 ,-3
670 DATA 0 , 0 ,-3 , 0
680 DATA 1 , 0 , 3 , 0
```

ANÁLISE

A linha 20 aciona as interrupções através das teclas de funcões (de F1 à F8).

A linha 30 define a variável N% (número de pontos) e a tela a ser usada.

As linhas de 40 à 90 iniciam algumas outras variáveis, simples e indexadas, do programa. Note que há uma matriz para cada coordenada espacial da figura (X,Y,Z) e uma matriz para o indicador ponto/traço (I). Há, ainda duas matrizes (S e T), apenas dimensionadas, para armazenar as coordenadas dos pontos na tela.

O laço entre as linhas 100 e 130 calcula, a partir das coordenadas espaciais (X,Y,Z), as coordenadas na tela dos pontos da figura. Essas coordenadas são armazenadas nas matrizes S e T.

O laço seguinte, de 140 a 170, efetiva o desenho na tela, isto é, ele constrói a figura.

As linhas 180 e 190 permitem controlar a figura a partir das teclas de funções (de F1 à F8). Cada uma dessas teclas desvia o programa para uma sub-rotina diferente. Essas sub-rotinas realizam transformações geométricas sobre os pontos da figura, produzindo rotações, deslocamentos e deformações. Não é nossa intenção explicá-las, apesar de serem bastante simples. Se você quiser entendê-las, procure em algum bom livro de álgebra linear a parte de transformações. Isso, entretanto, pressupõe conhecimentos a nível de segundo grau.

Quatro parâmetros do programa merecem ser ressaltados, pois são fundamentais para a movimentação das figuras. Dois deles, o S e o C, são definidos na linha 60 e determinam de quanto em quanto a figura gira quando a tecla F1, F2, F3 ou F4 é pressionada. Inicialmente usamos S=SIN(.2) e C=COS(.2). Experimente fazer S=SIN(.1) e C=COS(.1). Você notará que assim a rotação da figura se torna menos acentuada. Outros valores podem ser experimentados para S e C, contanto que se respeite a condição: S*S + C*C = 1 .

Os outros dois parâmetros, J% e K%, são definidos na linha 50 e determinam respectivamente a variação da distância e do efeito de perspectiva quando a tecla F5, F6, F7 ou F8 é usada. Como exemplo, usamos J%=5 e K%=1. Experimente fazer, J%=1 e K%=.9 .

Outro parâmetro, o fator de escala E% também pode ser alterado. Usamos E%=100. Teste E%=50 e E%=400. Deve ficar claro porque chamamos E% de fator de escala.

Agora que já explicamos o programa, daremos duas sequências de pontos que geram a projeção de um tesseracto (fig. 15.8) e uma multi-cruz (fig. 15.9), mas antes, introduza os dados necessários para a geração do cubo na tela. Lembre-se de alterar o valor de N% na linha 30 para cada figura que você introduzir!

DADOS DO CUBO:
N% = 16

```
560 ' Dados =)(p/t,x,y,z)
570 DATA 0 , 1 , 1 , 1
580 DATA 1 , 1 , 1 ,-1
590 DATA 1 , 1 ,-1 ,-1
600 DATA 1 , 1 ,-1 , 1
610 DATA 1 , 1 , 1 , 1
615 '
620 DATA 1 ,-1 , 1 , 1
630 DATA 1 ,-1 , 1 ,-1
640 DATA 1 ,-1 ,-1 ,-1
650 DATA 1 ,-1 ,-1 , 1
660 DATA 1 ,-1 , 1 , 1
665 '
670 DATA 0 ,-1 ,-1 ,-1
680 DATA 1 , 1 ,-1 ,-1
685 '
690 DATA 0 ,-1 ,-1 , 1
700 DATA 1 , 1 ,-1 , 1
710 '
720 DATA 0 ,-1 , 1 ,-1
730 DATA 1 , 1 , 1 ,-1
```

DADOS DO TESSERACTO:
N% = 36

```
560 ' Dados =)(p/t,x,y,z)
570 DATA 0 ,-1 ,-1 ,-1
580 DATA 1 , 1 ,-1 ,-1
590 DATA 1 , 1 ,-1 , 1
600 DATA 1 ,-1 ,-1 , 1
610 DATA 1 ,-1 ,-1 ,-1
615 '
620 DATA 1 ,-1 , 1 ,-1
630 DATA 1 , 1 , 1 ,-1
640 DATA 1 , 1 , 1 , 1
650 DATA 1 ,-1 , 1 , 1
660 DATA 1 ,-1 , 1 ,-1
665 '
670 DATA 1 ,-2 , 2 ,-2
```

```
680 DATA 1 , 2 , 2 ,-2
690 DATA 1 , 2 , 2 , 2
700 DATA 1 ,-2 , 2 , 2
710 DATA 1 ,-2 , 2 ,-2
715 '
720 DATA 1 ,-2 ,-2 ,-2
730 DATA 1 , 2 ,-2 ,-2
740 DATA 1 , 2 ,-2 , 2
750 DATA 1 ,-2 ,-2 , 2
760 DATA 1 ,-2 ,-2 ,-2
765 '
770 DATA 1 ,-1 ,-1 ,-1
780 DATA 0 , 1 , 1 ,-1
785 '
790 DATA 1 , 2 , 2 ,-2
800 DATA 1 , 2 ,-2 ,-2
810 DATA 1 , 1 ,-1 ,-1
820 DATA 1 , 1 , 1 ,-1
830 DATA 0 , 1 , 1 , 1
835 '
840 DATA 1 , 2 , 2 , 2
850 DATA 1 , 2 ,-2 , 2
860 DATA 1 , 1 ,-1 , 1
870 DATA 1 , 1 , 1 , 1
875 '
880 DATA 0 ,-1 , 1 , 1
890 DATA 1 ,-2 , 2 , 2
900 DATA 1 ,-2 ,-2 , 2
910 DATA 1 ,-1 ,-1 , 1
920 DATA 1 ,-1 , 1 , 1
```

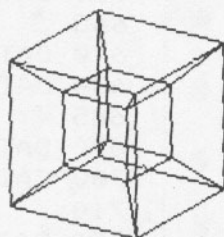



fig. 15.8

DADOS DA MULTI-CRUZ
N% = 48

```
560 ' Dados =)(p/t,x,y,z)
570 DATA 0 ,-1 , 1 , 3
580 DATA 1 , 1 , 1 , 3
590 DATA 1 , 1 ,-1 , 3
600 DATA 1 ,-1 ,-1 , 3
610 DATA 1 ,-1 , 1 , 3
620 '
```

```
630 DATA 1 ,-1 , 1 ,-3
640 DATA 1 , 1 , 1 ,-3
650 DATA 1 , 1 ,-1 ,-3
660 DATA 1 ,-1 ,-1 ,-3
670 DATA 1 ,-1 , 1 ,-3
680 '
690 DATA 0 , 3 , 1 , 1
700 DATA 1 , 3 , 1 ,-1
710 DATA 1 , 3 ,-1 ,-1
720 DATA 1 , 3 ,-1 , 1
730 DATA 1 , 3 , 1 , 1
740 '
750 DATA 1 ,-3 , 1 , 1
760 DATA 1 ,-3 , 1 ,-1
770 DATA 1 ,-3 ,-1 ,-1
780 DATA 1 ,-3 ,-1 , 1
790 DATA 1 ,-3 , 1 , 1
800 '
810 DATA 0 ,-1 , 3 , 1
820 DATA 1 ,-1 , 3 ,-1
830 DATA 1 , 1 , 3 ,-1
840 DATA 1 , 1 , 3 , 1
850 DATA 1 ,-1 , 3 , 1
860 '
870 DATA 1 ,-1 ,-3 , 1
880 DATA 1 ,-1 ,-3 ,-1
890 DATA 1 , 1 ,-3 ,-1
900 DATA 1 , 1 ,-3 , 1
910 DATA 1 ,-1 ,-3 , 1
920 '
930 DATA 0 , 1 , 1 , 3
940 DATA 1 , 1 , 1 ,-3
950 '
960 DATA 0 , 1 ,-1 , 3
970 DATA 1 , 1 ,-1 ,-3
980 '
990 DATA 0 ,-1 ,-1 , 3
1000 DATA 1 ,-1 ,-1 ,-3
1010 '
1020 DATA 0 , 1 , 3 , 1
```

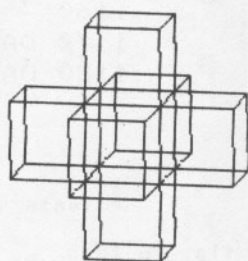



fig. 15.9

```
1030 DATA 1 , 1 ,-3 , 1
1040 '
1050 DATA 0 , 1 , 3 ,-1
1060 DATA 1 , 1 ,-3 ,-1
1070 '
1080 DATA 0 ,-1 , 3 ,-1
1090 DATA 1 ,-1 ,-3 ,-1
1100 '
1110 DATA 0 , 3 , 1 ,-1
1120 DATA 1 ,-3 , 1 ,-1
1130 '
1140 DATA 0 , 3 ,-1 ,-1
1150 DATA 1 ,-3 ,-1 ,-1
1160 '
1170 DATA 0 , 3 ,-1 , 1
1180 DATA 1 ,-3 ,-1 , 1
```

Tente você mesmo definir esta figura:

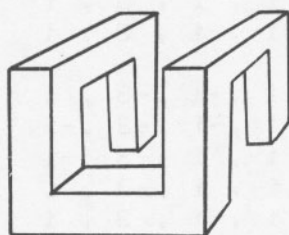
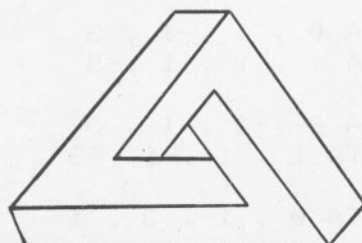
fig. 15.10

Ou esta:

fig. 15.11

PICASSO

INSTRUÇÕES

A pintura (rupestre) é a mais antiga forma de comunicação "escrita".

A arte evoluiu tecnicamente até um bom nível na época dos gregos e romanos, de certa forma regrediu durante a Idade Média e teve um novo impulso a partir do Renascimento. As técnicas foram sendo aperfeiçoadas e a arte figurativa atingiu requintes de perfeição.

Uma pintura podia parecer a reprodução de uma foto colorida. Daí o nascimento da Arte Moderna que se prende apenas a sentimentos e não a coisas existentes na natureza ou ao seu redor.

No programa Picasso a sua tela é oferecida à sua imaginação. Antes de rodá-lo, acione CAPS LOCK (trava de maiúsculas).

Ao ser executado, o programa gera, no centro da tela, um lápis que poderá ser conduzido pelas setas ou pelo joystick a qualquer ponto da tela.

Para desenhar um ponto aperte a barra de espaços ou o disparador do joystick.

O comando "D" desenha na parte inferior esquerda da tela um mostruário de cores.

O comando "A" apaga a tela e desenha o mostruário de cores.

O comando "P" pinta toda a região ao redor do lápis até encontrar uma borda da mesma cor.

O comando "DELETE" apaga os arredores da ponta do lápis.

O comando "9" é utilizado para mudar a cor do lápis. Após colocar a ponta do lápis em cima da cor desejada no mostruário, pressione a tecla "9".

O comando "C", na primeira vez que é acionado,

define o centro da circunferência e, na segunda, traça uma circunferência com o raio do centro (já definido) até a posição atual do lápis.

O comando "L", na primeira vez, define a extremidade de uma linha. Seguindo este comando existem outros três : L,B,F

L traça uma linha de um ponto já definido até o lápis.

B traça um quadro vazio, tendo como um dos vértices o primeiro ponto definido e, como outro, oposto ao primeiro, o ponto em que está o lápis.

F é como B, só que o quadro é pintado.

O comando "2" traça linhas com origem no último ponto definido com "L" ou "1" até o lápis.

O comando "1" traça linhas com origem no último ponto definido até o lápis e redefine o ponto na posição atual do lápis.

Agora deixe sua imaginação voar e mãos à obra.

DIGITAÇÃO

As linhas de 320 a 400, de 470 a 570 e de 590 a 660 podem ser digitadas como foi explicado no capítulo 1.

O programa é composto de 70 linhas.

```
10 'Picasso
20 COLOR 15,1,1:SCREEN 2
30 DRAW"C15S6BM168,175E7D7UH3GFBUBR6EU3
HGD6FRNE3R4NU6R2U3R3D3NL3BU2R"
40 DRAW"BR8BU3R4D8L4U4NR4U4BR7NR4D4R3FD
3GL3S4"
50 CIRCLE (128,88),80,13
60 LINE(0,0)-(50,50),7,B
70 LINE(9,9)-(41,41),5,BF
80 FOR F=60 TO 190:PSET(F,88),11:NEXT F
90 CIRCLE (128,88),30,11:PAINT(129,87),
11
100 FOR F=60 TO 191 STEP 2
110 LINE(128,168)-(F,88),11:NEXT F
120 FOR F=38 TO 66 STEP 7
130 CIRCLE(128,88),F,11
140 NEXT F
150 FOR F=0 TO 750:NEXT F
160 SCREEN 0
170 CLS:PRINT"                PICASSO"
180 PRINT:PRINT" ESCOLHA:"
```

```
190 PRINT:PRINT"(0) TECLADO"
200 PRINT"(1) JOYSTICK"
210 LOCATE0,7: LINE INPUT"OPCAO =>";A$
220 IF VAL (A$)>1 OR VAL (A$)<0 THEN 21
0
230 IF VAL (A$)>0 THEN J=1 ELSE J=0
240 SCREEN2,,0:P$=""
250 FOR F=1 TO 8
260 READ X$:T$=CHR$(VAL("&B"+X$))
270 P$=P$+T$
280 NEXT F
290 SPRITE$(1)=P$
300 X=128:Y=88:C=13
310 PUT SPRITE 0,(X,Y),C,1
320 IF STICK(J)=0 THEN 450
330 IF STICK(J)=1 THEN Y=Y-1
340 IF STICK(J)=5 THEN Y=Y+1
350 IF STICK(J)=3 THEN X=X+1
360 IF STICK(J)=7 THEN X=X-1
370 IF STICK(J)=2 THEN X=X+1:Y=Y-1
380 IF STICK(J)=4 THEN X=X+1:Y=Y+1
390 IF STICK(J)=6 THEN X=X-1:Y=Y+1
400 IF STICK(J)=8 THEN X=X-1:Y=Y-1
410 IF X>255 THEN X=0
420 IF Y>191 THEN Y=0
430 IF Y<0 THEN Y=191
440 IF X<0 THEN X=255
450 IF STRIG(J)=-1 THEN PSET(X,Y),C
460 I$=INKEY$:IF I$="" THEN 310
470 IF I$="0" THEN PLAY"A":GOSUB 670
480 IF I$="A" THEN PLAY"B":CLS:GOSUB 67
0
490 IF I$="C" THEN IF CI=0 THEN PLAY"C"
:CX=X:CY=Y:CI=1 ELSE PLAY"D":R=(CX-X)^2
+(CY-Y)^2:CIRCLE (CX,CY),SQR(R),C:CI=0
500 IF I$="L" THEN IF LI=0 THEN LX=X:LY
=Y:LI=1 ELSE LI=0:LINE(LX,LY)-(X,Y),C
510 IF I$="B" AND LI=1 THEN LI=0:LINE(L
X,LY)-(X,Y),C,B
520 IF I$="F" AND LI=1 THEN LI=0:LINE(L
```

```
X,LY)-(X,Y),C,BF
530 IF I$="P" THEN PAINT(X,Y+1),C
540 IF I$="2" THEN LINE(LX,LY)-(X,Y),C
550 IF I$="1" THEN LINE(LX,LY)-(X,Y),C:
LX=X:LY=Y
560 IF I$=CHR$(127) THEN LINE(X-1,Y)-(X
,Y+1),1,BF
570 IF I$="9" AND POINT(X+1,Y+1)<>1 THE
N C=POINT(X+1,Y+1)
580 GOTO 310
590 DATA 10000000
600 DATA 01100000
610 DATA 01110000
620 DATA 00101000
630 DATA 00010100
640 DATA 00001010
650 DATA 00000111
660 DATA 00000010
670 FOR F=0 TO 15
680 G=F*8:LINE(G,188)-(G+4,191),F,BF
690 NEXT F
700 RETURN
```

ANÁLISE

Na linha 20 definimos a cor e a tela que utilizaremos.

Nas linhas 30 e 40 fazemos a assinatura do autor e o ano.

Na linha 50 fazemos um círculo no centro da tela com raio 80 e cor 13.

Nas linhas 60 e 70 desenhamos dois quadrados sendo o primeiro vazio e o outro cheio.

Na linha 80 traçamos um risco da coordenada x=60 até x=190, ponto a ponto.

Na linha 90 desenhamos uma circunferência de raio 30 e pintamos metade dela.

As linhas de 100 à 110 traçam várias linhas com origem na parte mais inferior da circunferência até a linha central.

As linhas de 120 à 140 fazem várias circunferências concêntricas de raios diferentes.

Na linha 150 executamos uma pausa.

Na linha 160 voltamos à tela de texto.

Da linha 170 à 200 são impressos no vídeo o nome

do programa e as opções do teclado e joystick.

Na linha 210 introduzimos a opção.

Na linha 220 verificamos se a opção existe.

Na linha 230 verificamos qual foi a opção.

Na linha 240 definimos a tela de alta resolução e neutralizamos o som do teclado.

Da linha 250 à 290 há a leitura dos dados para o sprite e sua definição.

Na linha 300 definimos as coordenadas X,Y e também a cor.

Na linha 310 desenhamos o sprite na tela.

Da linha 320 à 400 há a leitura do teclado (joystick) com o movimento do lápis.

Da linha 410 à 440 verificamos a posição do sprite.

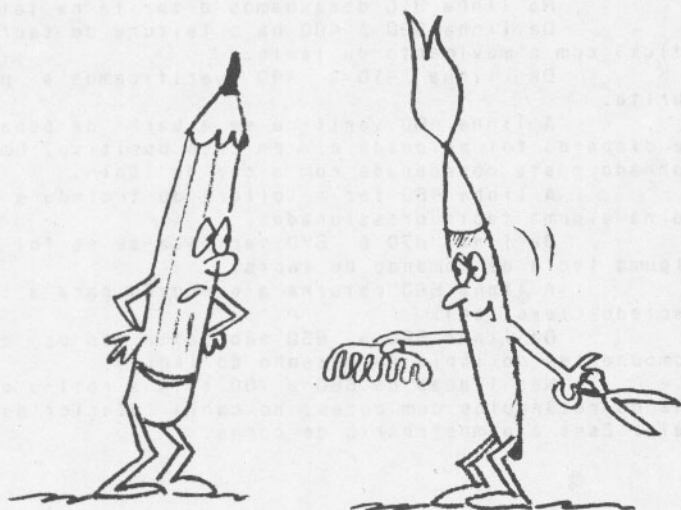
A linha 450 verifica se a barra de espaço (botão de disparo) foi acionada e, em caso positivo, um ponto é marcado nesta coordenada com a cor do lápis.

A linha 460 faz a leitura do teclado e verifica se há alguma tecla pressionada.

Da linha 470 à 570 verifica-se se foi acionada alguma tecla de comando do lápis.

A linha 580 retorna a execução para a leitura do teclado (joystick).

Da linha 590 à 660 são definidos os caracteres componentes do lápis (o desenho do lápis).

Nas linhas de 680 a 700 está a rotina que desenha os retângulos com cores, no canto inferior esquerdo da tela. Esse é o mostruário de cores.

CRAB CANON

INSTRUÇÕES

Johan Sebastian Bach, nascido no ano de 1685 em Eisenach, Saxônia, e falecido em Leipzig aos 75 anos, jamais imaginaria que sua música pudesse, um dia, ser executada com precisão acima de qualquer crítica por uma máquina. Ainda mais que qualquer pessoa, mesmo sem saber tocar qualquer instrumento, pudesse ler e transcrever rapidamente suas partituras para que esta máquina pudesse executá-las.

O programa que apresentamos é a transcrição de um pequeno canon de Bach para a notação musical cifrada do MSX. Com muito propósito, ele tem o nome de CRAB-CANON (Canon do Caranguejo). Note que a sua partitura possui uma fantástica simetria, envolvendo duas vozes! Experimente observá-la num espelho.

Se você estiver interessado nessa e em outras peculiaridades do Crab-Canon de Bach, procure ler (infelizmente em inglês, pois não há tradução!) o não menos fantástico livro de Douglas R. Hofstadter: GOEDEL, ESCHER, BACH: AN ETERNAL GOLDEN BRAID.

Para usar o programa, é só comandar RUN e RETURN (ou F5) e regular o volume do monitor de som!

Aos ouvidos mais exigentes, uma dica: ligue a saída AUDIO do Expert à entrada AUX de seu amplificador e regule o som da forma que você quiser. A diferença entre o som assim obtido e o do monitor é bastante significativa.

DIGITAÇÃO

Uma única letra (ou símbolo) incorreta ou não introduzida será o suficiente para transformar o CRAB-CANON num CRACK-CANNON e deixar seu ouvinte muito CRABBY, portanto, muita atenção!!!

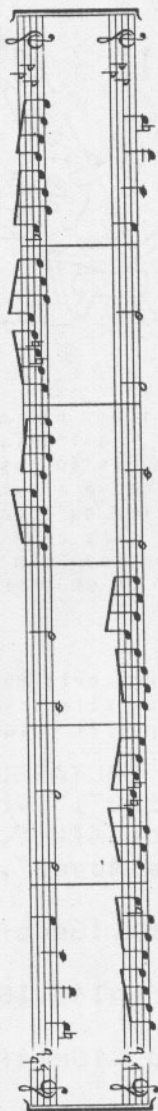
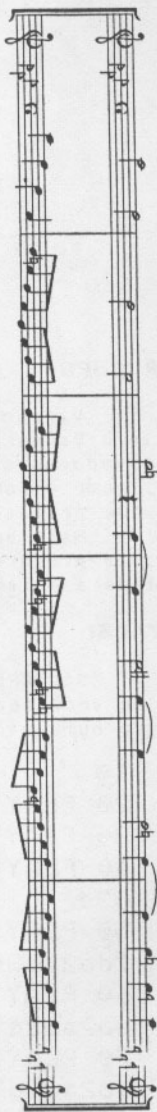
```
10 '        CRAB-CANON
20 PLAY"v10t120","v9t120"
30 PLAY"o312ce-gf","o314ce-go4cl8o3bo4c
de-fe-dc"
40 PLAY"o2l12br4o312gf#","o418do3go3dfe
-dco3babo4ce-"
50 PLAY"o312fee-d","o418dco3baga-b-o4d-
co3b-a-gfgfga-a-"
60 PLAY"o314dd-co2bgo3ce-","o318b-gfe-d
c-fgfe-db-"
70 PLAY"o312e-dce-","o318gfe-o4co3bagfe
-de-go4co3gfg"
80 PLAY"o318gfgo4co3ge-de-fgabo4co3e-fg
","o312e-cde-"
90 PLAY"o318a-de-fgfe-de-fga-b","o314fc
o2gbo3cd-"
100 PLAY"o318b-a-gfga-b-o4cd-o3b-a-gabo
4d","o312e-eff#"
110 PLAY"o418e-co3babo4cde-fdo3go4dcde-
f","o314ggr4o2bbo3aa"
120 PLAY"o418e-dco3bo4co3ge-c","o312ge-
c"
```

ANÁLISE.

Todas as linhas do programa têm a mesma função: gerar sons musicais através de dois canais independentes. A sintaxe usada da instrução PLAY em todas as linhas foi:

PLAY"sub-comandos musicais da primeira voz",
"sub-comandos musicais da segunda voz"

Os sub-comandos à esquerda da vírgula são executados através do canal 1 e, independentemente, os sub-comandos à direita da vírgula são executados através do canal 2.

CRAB CANON JSB

SALTARELLO

INSTRUÇÕES

Vincenzo Galilei, pai do grande Galileo, viveu durante o Renascimento, na Itália, e foi um dos grandes incentivadores da ópera, então nascente. Sua música Saltarello, como o nome sugere, é alegre e expontânea, transmitindo com precisão o clima da época em que foi feita.

Mais uma vez, para usar o programa, basta executá-lo, e preferencialmente, com a saída ÁUDIO do Expert conectada à entrada AUX de um amplificador.

DIGITAÇÃO

SALTARELLO é uma bela música para deixá-lo alegre. Se você esquecer ou alterar alguma nota, pode transformá-la numa marcha fúnebre! Cuidado ao digitar o programa!

```
10 '              SALTARELLO
20 PLAY"v10t180","v9t180"
30 PLAY"o4l8deL4f#g","l4o1do2ao3d"
40 PLAY"o4l2al4bgea","l4o1do2ao3do1do2a
o3d
50 PLAY"o4l2f#.l8edef#ge","o1l4do2ao3do
1do2ao3d"
60 PLAY"o4l2f#gl4e.l8f#","o1l4do2ao3do1
do2ao3d"
70 PLAY"o4l2d..l8el4f#g","l4o1do2ao3do1
do2ao3d"
80 PLAY"o4l4aabgea","l4o1do2ao3do1do2ao
3d"
```

```
90 PLAY"o4l2f#.l8edef#ge","l4oido2ao3do
ido2ao3d"
100 PLAY"o4l8f#f#gf#edc#o3bo4c#def#","l
4oido2ao3doido2ao3d"
110 PLAY"o4l2d.l4f#ed","l4oido2ao3doido
2ao3d"
120 PLAY"o4lif#l4ed","l4oiDo2ao3DoiDo2a
o3d"
130 PLAY"o4l2a.l4f#ed","l4oido2ao3doido
2ao3d"
140 PLAY"o4l2agl8eeef#","l4oido2ao3doid
o2ao3d"
150 PLAY"o4l2d.l8gf#l4ed","l4oido2ao3do
ido2ao3d"
160 PLAY"oo4l2f#.l8gf#l4ed","l4oido2ao3
doido2ao3d"
170 PLAY"o4l2a.l8edef#ge","l4oido2ao3do
ido2ao3d"
180 PLAY"o4l8f#f#ef#gf#edc#def#","oil4d
o2ao3doido2ao3d"
190 PLAY"o4l2d.l8o3del4f#g","l4oido2ao3
doido2ao3d"
200 PLAY"o3l4aabgea","l4oido2ao3doido2a
o3d"
210 PLAY"o3l2f#.l8edef#ge","l4oido2ao3d
oido2ao3d"
220 PLAY"o3l8f#f#gf#edc#o2bo3c#def#","l
ioid."
230 PLAY"o3l2d.l4f#ed","l4oido2ao3doido
2ao3d"
240 PLAY"o3l2f#.l4f#ed","l4oido2ao3doid
o2ao3d"
250 PLAY"o3l2a.l4f#ed","l4oido2ao3doido
2ao3d"
260 PLAY"o3l4al8gf#edc#o2bo3c#def#","li
oid."
270 PLAY"o3l2d.o4l8gf#l4ed","l4oido2ao3
doido2ao3d"
280 PLAY"o4l2f#.l8gf#ef#dd","l4oido2ao3
doido2ao3d"
```

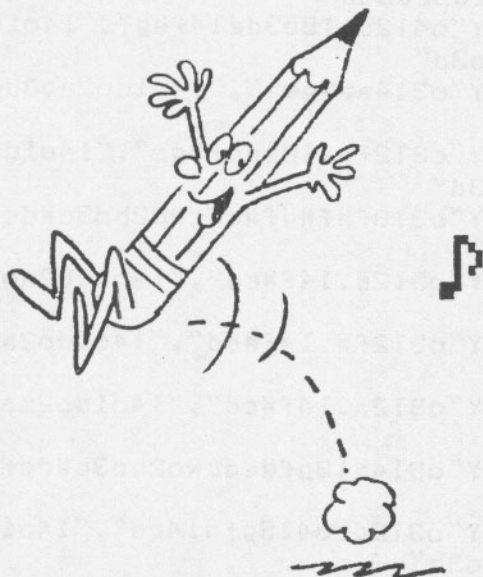
```
290 PLAY"o4l2a.l8gf#ef#ge","l4o1do2ao3d
o1do2ao3d"
300 PLAY"o4l8f#gaf#gagf#edef#","o1l4do2
ao3do1do2ao3d"
310 PLAY"l1d","l4o1do2ao3do1d"
```

ANÁLISE

Dois canais de som são utilizados. Todos os subcomandos sonoros à esquerda da vírgula são executados pelo canal 1 e, os que estão à direita da vírgula, pelo canal 2.

Experimente alterar a linha 20, deixando-a assim:

```
20 PLAY"v15t255m5000s0","v11t255m5000s0"
```

Dessa forma, o som ficará semelhante ao de um piano elétrico. Com algumas tentativas, você poderá conseguir sons mais sofisticados. Para dominar melhor o comando PLAY, consulte o livro LINGUAGEM BASIC MSX a partir da página 119.

INSTRUÇÕES

Os seis programas aqui apresentados geram efeitos sonoros usando o Gerador de Som Programável (PSG) do MSX.

Você poderá usá-los como sub-rotinas para implementar jogos ou sinalizar programas utilitários.

Digite-os, um de cada vez, e comande RUN. Para ouví-los com maior qualidade, você pode também ligar o Expert a um amplificador.

DIGITAÇÃO

Não há dificuldade em digitar nenhum dos seis programas. Todos são bastante curtos e simples.

ATERRISSAGEM

```
10 SOUND 7,56:SOUND 8,15:SOUND 9,0:SOUN
D 10,0:FOR L=0 TO 511:FOR M=L TO L+50 S
TEP 8:SOUND 0,M MOD 256:SOUND 1,M\256:N
EXT M,L
20 SOUND 7,7:SOUND 8,16:SOUND 12,32:SOU
ND 6,0:SOUND 13,15:FOR I=1 TO 500:NEXT
I:SOUND 13,0:END
```

CAMPAINHA

```
10 DATA 100,125,112,170,170,112,100,125
20 RESTORE:SOUND 7,56:SOUND 8,16:SOUND
```

```
1,0:SOUND 12,36:FOR I=1 TO 8:READ A:SOU
ND 0,A:SOUND 13,0:FOR L=1 TO 400:NEXT L
:IF I=4 THEN FOR L=1 TO 400:NEXT L:NEXT
 I ELSE NEXT I:END
```

ALARME 1

```
10 SOUND 7,56:SOUND 8,15:SOUND 1,0
20 SOUND 0,50:FOR I=1 TO 200:NEXT I:SOU
ND 0,100:FOR I=1 TO 200:NEXT I:GOTO 20
```

ALARME 2

```
10 SOUND 7,56:SOUND 8,15:SOUND 1,0
20 FOR L=230 TO 100 STEP -.5:SOUND 0,L:
NEXT L:SOUND 0,0:FOR I=1 TO 150:NEXT I:
GOTO 20
```

SIRENE

```
10 SOUND 7,56:SOUND 8,15:SOUND 1,0:SOUN
D 9,15:SOUND 3,0
20 SOUND 0,190:SOUND 2,192:FOR I=1 TO 4
00:NEXT I:SOUND 0,250:SOUND 2,253:FOR I
=1 TO 400:NEXT I:GOTO 20
```

DECOLAGEM

```
10 SOUND 7,28:SOUND 1,0:SOUND 3,0:SOUND
 6,0:SOUND 0,100:SOUND 8,13:SOUND 9,0:S
OUND 10,15:SOUND 12,255:FOR L=1 TO 2000
:NEXT L
20 FOR L=100 TO 30 STEP -.04:SOUND 0,L:
NEXT L:SOUND 8,0:FOR L=2 TO 31 STEP .01
:SOUND 6,L:NEXT L:SOUND 10,16:SOUND 13,
0
```

ANÁLISE

A programação do Gerador de Som Programável do MSX é feita de forma direta através do BASIC. A instrução usada é o SOUND, e para entender melhor como ela funciona, consulte o livro LINGUAGEM BASIC MSX, a partir da página 144.

C O L E Ç Ã O   M S X

* LINGUAGEM BASIC MSX - Denise Santoro Cruz
  Uma "enciclopédia" com todos os comando e instruções do BASIC MSX em ordem alfabética, explicados com programas exemplos.
  (outubro de 1985)

* DOMINANDO O EXPERT - Denise Santoro Cruz
  Os primeiros passos no BASIC MSX, explicados de maneira didática especialmente para os principiantes.
  (outubro de 1985)

* APROFUNDANDO-SE NO MSX - Piazzi-Maldonado-Oliveira
  Os segredos do VDP, PSG, PPI e do microprocessador do MSX explicados com dezenas de exemplos de forma didática para quem quer produzir programas para seu micro.
  (abril de 1986)

* COLEÇÃO DE PROGRAMAS PARA MSX VOL.I - Oliveira et al
  Programas de jogos, desenhos e músicas comentados detalhadamente para quem quer aprender a programar se divertindo.
  (fevereiro de 1986)

* COLEÇÃO DE PROGRAMAS PARA MSX VOL.II - Oliveira et al
  Programas didáticos, jogos, aplicativos e utilitários, comentados e explicados passo a passo, com truques e dicas de programação.
  (junho de 1986)

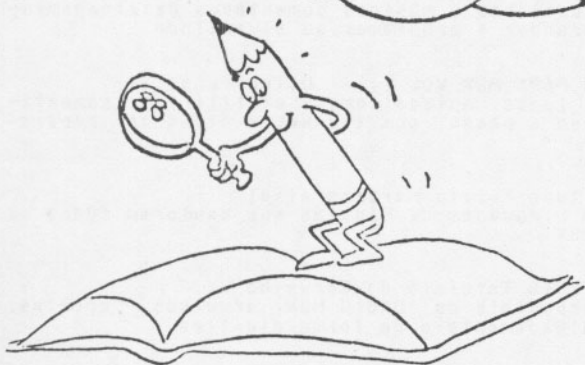
* JOGOS PARA MSX - Wilson Fazzio Martins et al
  Jogos em BASIC e em Linguagem de Máquina que exploram todos os recursos do padrão MSX.
  (julho de 1986)

* EXPLORANDO O MSX - Luis Tarcísio de Carvalho Jr.
  As características especiais do BASIC MSX: arquivos, sprites, etc, explicados detalhadamente e de forma didática.
  (agôsto de 1986)

* USANDO ASSEMBLY NO MSX - Maldonado et al
  Como usar linguagem de máquina no MSX, com exemplos e aplicações práticas.
  (setembro de 1986)

* PROGRAMAÇÃO AVANÇADA NO MSX - Rossini
  Aprendendo a programar em linguagem de máquina no MSX, com exemplos e aplicações práticas.
  (setembro de 1986)

Se você quiser receber
gratuitamente nosso boletim
informativo com dicas e programas
para seu MSX, destaque
o cupom abaixo (ou uma
xerox dele) e envie-o para

ALEPH PUBLICAÇÕES E
ASSESSORIA PEDAGÓGICA LTDA.
Av. Brig. Faria Lima, 1451- Cj.31
01451 - S.Paulo - S.P.
Cx.Postal, 20707 Tel, (011) 212-4917

NOME: ....
END.: ....
BAIRRO: .... CEP: ....
CIDADE: .... U.F: ....

O texto deste livro foi composto numa impressora MÔNICA PLUS da ELEBRA, utilizando caracteres residentes e fontes alternativas desenvolvidas pela ELEBRA. As amostras de tela foram obtidas colocando-se a MÔNICA no modo gráfico e utilizando-se rotinas de cópia para MSX desenvolvidas pela equipe de programação da ALEPH, publicadas nos livros APROFUNDANDO-SE NO MSX, COLEÇÃO DE PROGRAMAS PARA MSX VOL. II e PROGRAMAÇÃO AVANÇADA NO MSX.

COLEÇÃO MSX

A implantação do padrão MSX no Brasil gerou uma grande procura de literatura específica por parte dos usuários. Por ser uma máquina recente em todo o mundo, mesmo as publicações estrangeiras são escassas. Para suprir as necessidades dos usuários nacionais, a EDITORA ALEPH vem produzindo uma série de livros e outros materiais dedicados a essa linha de micros, que tem tudo para se tornar uma das mais difundidas em nosso país.

Com o característico zêlo pela clareza e inteligibilidade de suas publições, mais uma vez saindo e chegando na frente, a EDITORA ALEPH brinda seus leitores com esta reedição de um dos maiores sucessos da "COLEÇÃO MSX".

Dificilmente o leitor conseguirá encontrar, mesmo em livros importados ou traduzidos, explicações tão didáticas, dicas e "macêtes" tão procurados como no "COLEÇÃO DE PROGRAMAS PARA O MSX", uma coletânea de quase 40 programas comentados passo a passo e que tem como finalidade última mostrar como se pode usar o poderosíssimo BASIC MSX!